An errata for this document is available.
See Document ID#: IMX51RMAD.

MCIMX51 Multimedia Applications Processor Reference Manual

MCIMX51RM
Rev. 1
2/2010

freescale semiconductor

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 010 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
+1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: MCIMX51RM
Rev. 1, 2/2010

Contents

Contents

Contents

Contents

Contents

Contents

Contents

Chapter 18
Configurable Serial Peripheral Interface (CSPI)

Chapter 19
Clock Amplifier (CAMP)

Contents

Contents

Contents

Contents

Contents

Contents

Chapter 32
Fast Ethernet Controller (FEC)

Contents

Contents

Contents

Contents

Chapter 39
High Speed Inter IC (HS-I^{2}C)

Contents

Chapter 40
Inter IC (I^2C)

Contents

Contents

Contents

Chapter 45
NAND Flash Controller (NFC)

Contents

Contents

Contents

Contents

Chapter 52
Smart Direct Memory Access (SDMA) Controller

Contents

Contents

Contents

Contents

Contents

Contents

Chapter 61
Video Processing Unit (VPU)

Contents

Contents

Contents

Appendix A
IOMUX Controller (IOMUXC)

About This Book

The primary objective of this reference manual is to define the functionality of the MCIMX51 (i.MX51). The i.MX51 is ideal for power-thirsty applications, such as video and audio media players. The i.MX51 processor includes leading power management, security management, digital rights management, video and image processing technology, providing a formidable combination of the features OEMs desire to drive high-performance video and audio content on wireless mobile devices. The processor is designed with Freescale's Smart Speed Technology, which enables ultra low-power consumption and performance equivalent to processors with much higher MHz.

Audience

It is assumed that the reader understands operating systems, microprocessor system design, and the basic principles of RISC processing.

Organization

Following is a summary and a brief description of the major parts of this reference manual:

Book I contains information that is unique to the i.MX51. The following chapters are included:

- Chapter 1, "Introduction" provides a high-level description of features and functionality of the integrated host processor. It describes the MCIMX51, its interfaces, and its programming model. The functional operation with emphasis on peripheral functions is also described.
- Chapter 2, "Memory Map" describes the memory map. An overview of the local address map is followed by a description of how local access windows are used to define the local address map. The inbound and outbound address translation mechanisms used to map to and from external memory spaces are described next. Finally, the configuration, control, and status registers are described, including a complete listing of all memory-mapped registers with cross references to the sections detailing descriptions of each.
- Chapter 3, "Interrupts and DMA Events" provides a listing of the Interrupts and DMA Events. This chapter also provides information on the assignments of interrupts and of the DMA events.
- Chapter 4, "External Signals and Pin Multiplexing" describes the external signals and pin multiplexing. The input-output multiplexer (IOMUX) controls the pin multiplexing. The IOMUX also configures other pin characteristics, such as voltage level, drive strength, and hysteresis. This chapter organizes the signals by functional group (as defined by the top-level block diagrams) and then by subgroups (interfaces) within each functional group.
- Chapter 5, "External Memories" describes the module that controls all i.MX51 external memory accesses (read/write/erase/program) from all the masters in the system to different external memories. All accesses are arbitrated by the Multi Master Multi Memory Interface (M4IF) module and controlled by the respective memory controller. Because different modules require different pad settings (like pull up, keeper, and others), the IOMUXC controls also the pad settings parameters.
- Chapter 6, "Fuse Map" lists all fusable elements in the i.MX51.

- Chapter 7, “Clock Controller Module (CCM)” controls the clocks for the i.MX51 modules. This module uses the available clock sources to generate the clock roots.
- Chapter 8, “Debug Architecture” describes the hardware and software debug and application development features and resources of i.MX51. There are core/platform-specific resources, resources associated with some of the more complex IP blocks, and chip-wide resources.
- Chapter 9, “System Boot” describes the system boot sequence of the i.MX51 application processor and provides the details of boot options.[1]
- Chapter 10, “Multimedia” contains descriptions of the major components of the Multimedia module including the Video Processing Unit (VPU), Graphics Processing Unit (GPU): accelerating 2D/3D graphics, Image Processing Unit (IPU): providing connectivity to cameras and displays, related processing, synchronization and control.
- Chapter 11, “Power Management” describes the operation of the various power management techniques supported by the i.MX51 and the registers used to configure and control them. These power management techniques reduce active and static power consumption.
- Chapter 12, “System Security” describes the security features designed into the i.MX51.

Book II contains chapters describing modules in the i.MX51 that are common to the MX generation of ICs. The following chapters are included:

- Chapter 13, “1-Wire Module (1-Wire)” describes the 1-Wire module, which provides the communication link to a generic 1-Kbit add-only memory. The module sends or receives one bit at a time with an option for software to manage the data using bytes.
- Chapter 14, “Cortex-A8 Platform” provides an overview of the ARM Cortex-A8n platform, which includes a NEON co-processor, an L1 cache, an L2 cache, an ETM and a CTI. The platform includes the essential sub-blocks: platform control, test logic and the debug modules (CTM, ETB, and a second CTI).
- Chapter 15, “Platform Debug” provides an overview of the ARM platform debug. It will cover the modules inside the tigerp_gp_debug block and also the modules that support the platform debug within the SOC, but external to the ARM platform
- Chapter 16, “Multi-Layer AHB Crossbar Switch (MAX)” provides an overview of the MAX (Multi-Layer AHB Crossbar Switch). The purpose of the MAX is to concurrently support up to 4 simultaneous connections between master ports and slave ports.
- Chapter 17, “Digital Audio Mux (AUDMUX)” describes the AUDMUX, which provides a programmable interconnect device for voice, audio, and synchronous data routing between host serial interfaces (that is, SSI, SAP) and peripheral serial interfaces (that is, audio and voice CODECs, also known as coder-decoders).
- Chapter 18, “Configurable Serial Peripheral Interface (CSPI)” describes the Configurable Serial Peripheral Interface (CSPI) module, which allows rapid data communication with fewer software interrupts than conventional serial communications.
- Chapter 19, “Clock Amplifier (CAMP)” describes the Clock Amplifier, which converts a square wave/sinusoidal input of frequency range 8–40 MHz into a rail-to-rail square wave (CAMP supply voltage).

1. Portions of this chapter are restricted. See your Freescale representative for details.

- Chapter 20, “Central Security Unit (CSU)” describes the Central Security Unit (CSU), which enables software for setting comprehensive security policy within the platform and sharing secure information between various secure modules.
- Chapter 21, “Digital Phase Lock Loop (DPLL)” describes the three on-chip Digital Phase Lock Loops (DPLLs), which provide clock generation in digital and mixed analog/digital chips designed for wireless communication and other applicationsThis chapter provides an overview of the DPLL opertation.
- Chapter 22, “DPLL-IP Interface (DPLL-IP)” describes the DPLL-IP, which serves as an interface to the DPLL module. It generates the control signals required for the DPLL operations. In other MX ICs this used to be part of the SRC (System Reset Controller) module
- Chapter 23, “Dynamic Process and Temperature Compensation (DPTC)” describes the DPTC (Dynamic Process and Temperature Compensation) module, which is a power management module. The purpose of the DPTC module is to detect the minimum operation voltage for the IC, regarding process corner case and temperature for a given frequency
- Chapter 24, “Dynamic Voltage Frequency Scaling (DVFS)” The DVFS allows simple dynamic voltage frequency scaling. The frequency of the core clock domain and the voltage of the core power domain can be changed on the fly while all modules (including the MCU) continue their normal operation.
- Chapter 25, “Dynamic Voltage Frequency Scaling for Peripherals (DVFS_PER)” Dynamic Voltage and Frequency Scaling (DVFS) enables the frequency of the system and the voltage of the power domain to be changed on the fly while all modules continue their normal operation on reduced frequency.
- Chapter 26, “Enhanced Configurable Serial Peripheral Interface (eCSPI)” The eCSPI module allows rapid data communication with fewer software interrupts than conventional serial communications methodologies.
- Chapter 27, “External Memory Interface (EMI)” The EMI is a memory controller of memory devices in the system, both internal and external. It provides a capability for the system to control the external memory device, by preforming several type of accesses like read, write, program, and erase as well as internal memories of the system.
- Chapter 28, “Embedded Memory Periphery Power Gating Controller (EMPGC)”
- Chapter 29, “Enhanced Periodic Interrupt Timer (EPIT)” The Enhanced Periodic Interrupt Timer (EPIT) is a 32-bit set-and-forget timer which begins counting after the EPIT is enabled by software. It is capable of providing precise interrupts at regular intervals with minimal processor intervention.
- Chapter 30, “Enhanced SDRAM Controller (eSDCTL)” The Enhanced SDRAM Controller version 2 (eSDCTLV2) consists of nine major blocks, including the SDRAM command state machine controller, bank register (page and bank address comparators), Row/Column Address Multiplexer, configuration registers, refresh request counter, command sequencer, size logic (splitting access), data path (data aligner/multiplexer), LPDDR interface, and the Power Down timer.
- Chapter 31, “Enhanced Secured Digital Host Controller (eSDHC)” The eSDHC provides the interface between the host system and MMC/SD/SDIO/CE-ATA cards, including cards with reduced size or mini cards.

- Chapter 32, “Fast Ethernet Controller (FEC)” This chapter provides a feature-set overview, a functional block diagram, and transceiver connection information for both the 10 and 100 Mbps Media Independent Interface (MII), as well as the 7-wire serial interface.
- Chapter 33, “Fast Infrared Interface (FIRI)” The Fast Infrared Interface module (FIRI) is capable of establishing any of the following links: 0.576 Mbit/s, 1.152 Mbit/s, or 4 Mbit/s half duplex (using a LED and IR detector).
- Chapter 34, “General Power Controller (GPC)” GPC module controls the following Advanced Power Saving features:DVFS, DPTC, and all of the SRPG modules.
- Chapter 35, “General Purpose Input/Output (GPIO)” The General Purpose Input/Output (GPIO) module provides 32 bits of bidirectional, general-purpose input and output signals.
- Chapter 36, “General Purpose Timer (GPT)” The General purpose timer (GPT) has a 32 bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge.
- Chapter 37, “Graphics Processing Unit 2D (GPU2D)” The GPU is an embedded 2D and vector graphics accelerator targeting the OpenVG 1.1 graphics API and feature set. It accelerates 2D bitmap graphics operations, such as BitBlt, fill and raster operations using a separate 2D graphics acceleration unit. Vector graphics rendering is accelerated by a separate anti-aliasing polygon rasterizer, which is connected to the 2D graphics acceleration unit.
- Chapter 38, “3D Graphics Accelerator (GPU3D)” The GPU3D (3D graphics processing unit) is based on the AMD Z430 (also known as ATI Yamato DX). The contains an embedded engine capable of DirectX9 Shader Model 3.0+ program execution. The module focus is on accelerating user level graphics APIs such as OpenGL ES 2.0 & 1.1, and Direct3D Mobile 1.2.
- Chapter 39, “High Speed Inter IC (HS-I^2C)” The High Speed Inter IC (HS-I^2C) is bi-directional, two wire, serial bus interface module. The HS_I^2C is designed to be compatible with the standard Philips I^2C bus protocol version 2.1.
- Chapter 40, “Inter IC (I^2C)” The Inter IC (I^2C) module provides functionality of a standard I^2C slave and master. The I^2C module is designed to be compatible with the standard Philips I^2C bus protocol.
- Chapter 42, “Image Processing Unit 3 (IPUv3EX)” The IPU is part of the video and graphics subsystem in the i.MX51. It provides comprehensive support for the flow of data from an image sensor and/or to a display device.
- Chapter 43, “Keypad Port (KPP)” The Keypad Port (KPP) is a 16-bit peripheral that can be used as a keypad matrix interface or as general purpose input/output (I/O).
- Chapter 44, “Multi Master Multi Memory Interface (i.MX51)” The Multi-Master Multi Memory Interface (M4IF) controls memory accesses (read/write/erase/program) from one or more masters through different port interfaces to different external memory controllers ESDCTL, NFC, and WEIM, as well as two internal memories in the system as well.
- Chapter 45, “NAND Flash Controller (NFC)” Composed of various control logic units, a 4.5-Kbyte internal RAM buffer and an internal ECC mechanism, the NAND Flash Controller (NFC) implements the interface to standard NAND Flash memory devices.

- Chapter 46, “Parallel Advanced Technology Attachment (P-ATA)” The ATA block is an AT attachment host interface. Its main use is to interface with hard disc drives and optical disc drives. It interfaces with the ATA device over a number of ATA signals.
- Chapter 47, “Pulse-Width Modulator (PWM)” The pulse-width modulator (PWM) has a 16-bit counter, and is optimized to generate sound from stored sample audio images and it can also generate tones. It uses 16-bit resolution and a 4×16 data FIFO to generate sound.
- Chapter 48, “Run-Time Integrity Checker (RTIC)” The Run-Time Integrity Checker (RTIC) function is to ensure the integrity of the peripheral memory contents, and assist with boot authentication. The RTIC has the ability to verify the memory contents during system boot and during run-time execution.
- Chapter 49, “Symmetric/Asymmetric Hashing and Random Accelerator (SAHARA)” The Symmetric/Asymmetric Hashing and Random Accelerator (SAHARA) is a security co-processor that can be used on cell phone baseband processors or wireless PDAs. It implements block encryption algorithms, (AES, DES, and 3DES), hashing algorithms (MD5, SHA-1, SHA-224, and SHA-256), stream cipher algorithm (ARC4), and a hardware random number generator.
- Chapter 50, “Security Controller (SCC)” The Security Controller (SCC) is composed of two sub-blocks, secure RAM and a security monitor.
- Chapter 51, “Subscriber Identification Module (SIM)” The Subscriber Identification Module (SIM) is designed to facilitate communication to SIM cards or Eurochip pre-paid phone cards. The SIM module has two ports that can be used to interface with the various cards.
- Chapter 52, “Smart Direct Memory Access (SDMA) Controller” The Smart Direct Memory Access (SDMA) controller offers highly-competitive DMA features combined with software-based virtual-DMA flexibility. It enables data transfers between peripheral I/O devices and internal/external memories.
- Chapter 53, “Sony/Philips Digital Interface Transmitter (SPDIF Tx)” The Sony/Philips Digital Interface Transmitter (SPDIF Tx) audio module is a stereo transmitter that allows the processor to transmit digital audio over it.
- Chapter 56, “Synchronous Serial Interface (SSI)” The SSI is a full-duplex, serial port that allows the chip to communicate with a variety of serial devices. These serial devices can be standard CODer-DECoder (CODECs), Digital Signal Processors (DSPs), microprocessors, peripherals, and popular industry audio CODECs that implement the inter-IC sound bus standard (I^2S) and Intel AC97 standard.
- Chapter 58, “TV Encoder (TVE)” The TV Encoder (TVE) is designed to provide direct connection between an Application Processor (AP) and a TV set via analog interfaces.
- Chapter 59, “y Universal Asynchronous Receiver/Transmitter (UART)” The UART supports serial RS-232 NRZ format, and IrDA.
- Chapter 60, “Universal Serial Bus OTG HOST (USBOH3)” The USBOH3 module contains all of the functionality required to support four independent USB ports which are compatible with the USB 2.0 specification. Three of the USB host controllers are identical. In addition to the normal USB functionality, the module also provides support for direct connections to on-board USB peripherals using serial or ULPI protocol.

- Chapter 61, “Video Processing Unit (VPU)” Video Processing Unit of the i.MX51 is a high performance multi-standard video processing unit which can perform H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG4 SP/ASP, Divx ,RV8/9,and MPEG2 MP decoding up to 1920x1088 resolution.
- Chapter 62, “Watchdog Timer (WDOG)” The Watchdog (WDOG) timer module protects against system failures by providing a method of escaping from unexpected events or programming errors.
- Chapter 63, “Wireless External Interface Module (WEIM)” The Wireless External Interface Module (IPU) handles the interface to devices external to the chip, including generation of chip selects, clock and control for external peripherals and memory. It provides asynchronous access to devices with SRAM-like interface and synchronous access to devices with NOR-Flash like or PSRAM like interface.
- Appendix A, “IOMUX Controller (IOMUXC)” The IOMUX controller (IOMUXC), together with the IOMUX, enables the IC to share one BGA contact with several functional blocks. The sharing is done by multiplexing the BGA contact input/output signals.

MCIMX51 Reference Manual
Book I

Rev. 1
2/2010

Chapter 1
Introduction

This chapter introduces the architecture of the MCIMX51 (i.MX51) Multimedia Applications Processor. The i.MX51 processor represents Freescale Semiconductor's latest achievement in multimedia integrated applications processors that are part of a growing family of multimedia-focused products offering high performance processing optimized for lowest power consumption.

A high-level description of features and functionality of the integrated host processor are provided in this chapter. The different modules are summarized and the interfaces are also described, along with the respective programming model. Functional operation with emphasis on peripheral functions is also included.

NOTE

Because the i.MX51 and i.MX51A are functionally identical, all references to the i.MX51 in this reference manual also apply to the i.MX51A.

1.1 Target Applications

The flexibility of the architecture allows it to be used in a wide variety of applications. Regardless of whether the customer is designing a smartphone, PDA, gaming console, portable media player (PMP), portable navigation device (PND), web tablets, or other portable device, the i.MX51 processor provides design with the power and flexibility necessary for today's competitive marketplace.

1.2 Features

The i.MX51 processor is based on an ARM Cortex A8™ platform, which has the following features:

- ARM Cortex A8™ Processor (with TrustZone)
- 32 Kbyte L1 Instruction Cache
- 32 Kbyte L1 Data Cache
- 256 Kbyte L2 cache
- Neon coprocessor
  - SIMD Media Processing Architecture
  - NEON register file with 32 × 64-bit general-purpose registers
  - NEON Integer execute pipeline (ALU, Shift, MAC)
  - NEON dual, single-precision floating point execute pipeline (FADD, FMUL)
  - NEON load/store and permute pipeline
  - Non-pipelined Vector Floating Point (VFP) co-processor (VFP)

The maximum frequency of the core is:

- 800 MHz (consumer version)
- 600 MHz (automotive and extended temperature version)

See respective data sheet for details about frequency, voltage, and temperature ranges.

To boost the multimedia performance, the following hardware accelerators are integrated:

- VPU—Video Processing Unit
- IPU—Image Processing Unit
- GPU 3D—Graphics Processing Unit (OpenGL ES 2.0)
- GPU 2D—Graphics Processing Unit, 2D (OpenVG 1.1)

Security functions are enabled and accelerated by the following hardware:

- ARM TrustZone including the TZ architecture (separation of interrupts, memory mapping, etc.)
- SJC—Secure JTAG Controller. Protecting JTAG from debug port attacks by regulating or blocking the access to the system debug features
- SRTC—Secure Real-Time Clock. Tamper resisted RTC with its own power domain and mechanism to detects voltage and clock glitches
- RTIC—Real-Time Integrity Checker. RTIC type 1, enhanced with SHA-256 engine
- SAHARA Lite—Cryptographic accelerator that includes true random number generator (TRNG)
- SCC—security controller type 2. Improved SCC with AES engine, Secure/Non-Secure RAM and support for multiple keys as well as TZ/non-TZ separation
- CSU—Central Security Unit. Enhancement for the IIM. Will be configured during boot and by e-fuses and will determine the security level operation mode as well as the TZ policy
- A-HAB—Advanced High Assurance Boot – HAB with the next embedded enhancements: SHA-256, 2048 bit RSA key, version control mechanism, warm boot, CSU and TZ initialization

The memory system consists of the following levels:

- Level 1 Cache
  - Instruction (32 Kbyte)
  - Data (32 Kbyte)
- Level 2 Cache
  - Unified instruction and data (256 Kbyte)
- Level 2 Memory
  - Boot ROM, including HAB (36 KB)
  - Unified Internal RAM (128 KB), including Secure/Non-Secure RAM. Support flexible allocation of secure/non-secure, at 8 KB blocks granularity. Controlled by SCC module.

The i.MX51 SoC is built around the following system buses:

- 64-bit AMBA AXI v1.0. It is referenced further as AXI and used by ARM Cortex A8™ Platform, major multimedia accelerators (VPU, IPUEX), and EMI.

- 32-bit AMBA AHB 2.0. It is referenced further as AHB and used by most of the bus master peripherals, such as SDMA, RTIC, SCC etc. (see block diagram for the complete list).
- 32-bit IP. It is used for control (and slow data traffic) of the most SoC peripheral devices.

The following interfaces are available for external devices (some are muxed and not available simultaneously):

- Hard Disk Drives
  - CE-ATA, up to 416 Mbps
  - P-ATA, up to 66 MByte/s
- Displays
  - Two display ports, each can drive displays using either (or both) of the following interfaces
    - Parallel interface: up to 24-bit data bus, up to 100 MHz
    - DSI fast serial interface: up to 2 data lanes, up to 1600 Mbps
  - NTSC/PAL TV-Out interface via integrated video encoder.
  - Pixel Color depth up to 24-bits
- Camera sensors
  - Two camera ports (each can use either parallel or serial interface)
    - Parallel Camera sensor (up to 16-bit, peak up to 120 Mpixels/sec)
- Expansion cards
  - Four SD/MMC cards , The MMC card supports operation of 416 Mbps in 8-bit mode. The SD/SDIO supports 1-bit and 4-bit modes up to 200 Mbps.
- External memory interfaces
  - 32-bit DDR2, mobile DDR - both with 200 MHz clock
  - 8/16-bit NAND SLC/MLC Flash, up to 50 MHz
  - 16-bit NOR Flash. 32-bit width is supported in multiplexed Address/Data mode. All WEIM pins are muxed on other interfaces (data with NFC pins). IO muxing logic selects WEIM port, as primary muxing at system boot.
  - PSRAM, Cellular RAM
  - MDOC NAND Flash and OneNAND
  - 32-bit is supported in Muxed mode
- USB
  - USB 2.0 OTG, up to 480 Mbps
  - Integrated High Speed USB Phy
  - 3 USB 2.0 Hosts, up to 480 Mbps each
    - Only OTG port supports overcurrent and PM signals.
  - External HS/FS Transceivers, (ULPI / Serial), at expense of functionality, due to IO muxing limitations.
- Low-Power Modes (LPM)
  - Supporting DVFS and DPTC techniques for low power modes

- Uses SRPG (State Retention Power Gating) for ARM and Neon
- Power-gating for VPU, GPU and Save & Restore (S&R) Power gating for IPU.
- Partial SRPG of EMI
- Support for various levels of system power modes
  - Peripherals are divided into 2 groups. One of the groups supports SRPG.
- Flexible clock gating control scheme

- Miscellaneous IPs and interfaces:
  - 1-Wire
  - Three (3) I^2S/SSI/AC97, up to 1.4 Mbps each
  - Three (3) UART, up to 4.0 Mbps each
    - One supports 8-bit and others supports 4-bit.
  - One CSPI
  - Two eCSPI, up to 52 Mbps each
  - One HS-I^2C, supports 3.4Mbps[1]
  - Two (2) I^2C, supports 400 kbps
  - Fast Ethernet Controller, 10/100 Mbps
  - Two Pulse Width Modulators (PWM)
  - JTAG Controller (SJC)
  - GPIO with interrupt capabilities
  - Key Pad Port (KPP)
  - One SIM, up to 33 Mbps
  - Sony Phillips Digital Interface Transmitter (SPDIF)
- Watchdog timers
- Audio MUX

1.3 Architectural Overview

This section contains a simplified block diagram of the i.MX51.

1. Not recommended for use in new designs. See i.MX51 IC errata

1.3.1 Block Diagram

Figure 1-1 shows a high-level block diagram of the i.MX51, providing a view of the major subsystems (processor domains, shared peripherals domain, memories, etc.) and logical connectivity.

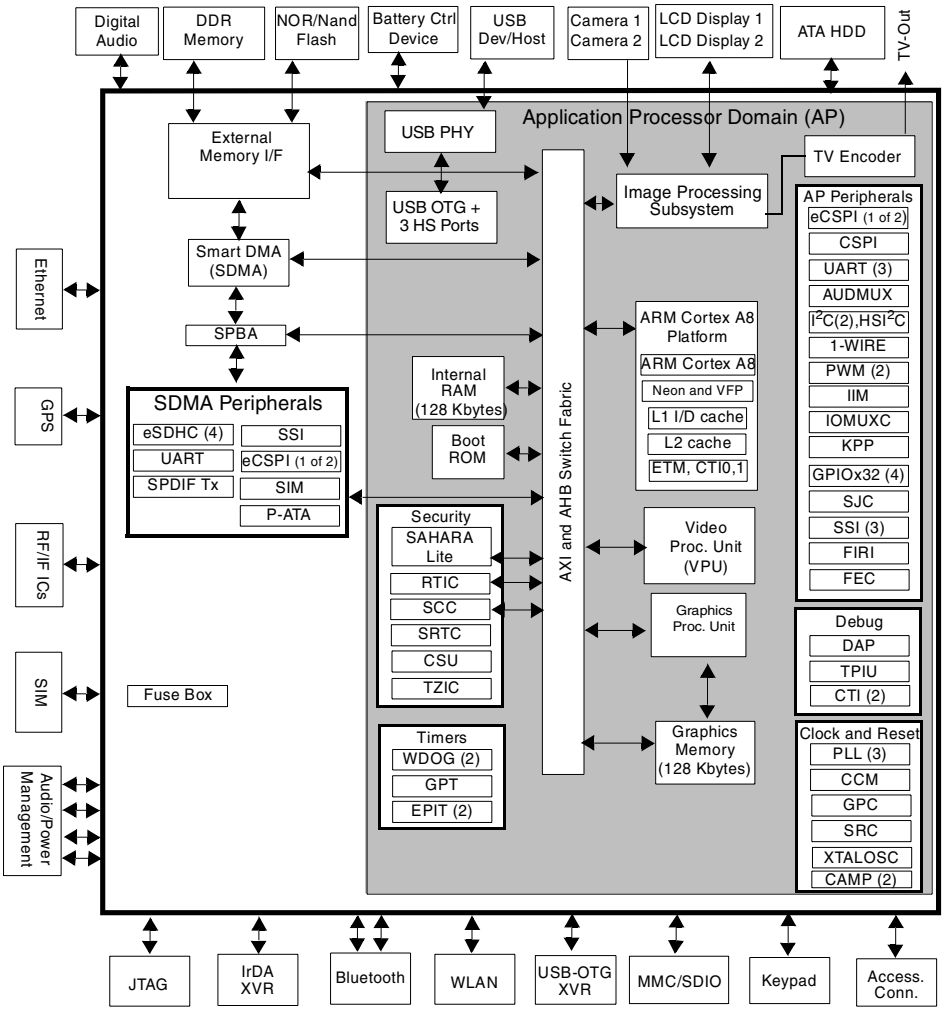


Figure 1-1. i.MX51 Simplified Block Diagram

1.3.2 Major Subsystems

The i.MX51 consists of the following major subsystems:

- Core (ARM Cortex A8™) Platform, L1/L2 memories
- SDMA controller and the shared peripheral domain
- System Control—Boot Flow control, Clocks distribution, “Reset” control and Low Power logic

- Multimedia
  - Video Processor
  - TV-Encoder (TVe-) for PAL/NTSC output
  - Image Processor
  - Graphics Processors
  - Audio—connectivity interfaces in hardware, while codecs are performed in SW by ARM core.
- Security
- Connectivity peripherals
- External Memory Interface

1.3.3 Architectural Partitioning

This paragraph defines how architecture supports the processing intensive tasks:

- ARM Cortex A8™ Platform is responsible for:
  - Operating System
  - User applications (including control over hardware accelerators and non-accelerated functions)
  - TrustZone applications
- Smart DMA enables data transfer between non-mastering peripherals and external or internal memories
- System Control is supported via:
  - Clock Control Block (CCM)
  - Three (3×) PLLs
  - XTALOSC—Crystal oscillator source support
  - Frequency Pre-multiplier (FPM)
  - System Reset Controller (SRC)
  - Global Power Controller (GPC)
  - Two (2×) CAMPs—Clock Amplifier modules on the CKIH1 and CKIH2 inputs
- Multimedia is supported with:
  - Image Processing Unit—IPUEX
    - Connectivity to displays, display controllers, and auxiliary graphics co-processors
    - Display Processing: video/graphics combining, image enhancement
    - Image conversions: resizing, rotation/inversion, color conversion
    - Synchronization and control capabilities, allowing autonomous operation
  - Video Processing Unit (VPU) (Video codecs, all in hardware unless mentioned otherwise):
    - MPEG-4 decode: 1280 × 720, 30 fps, Simple Profile and Advanced Simple Profile
    - MPEG-4 encode: D1, Simple Profile
    - H.263 decode: 1280 × 720, 30 fps, Profile 3
    - H.263 encode: D1, Profile 3

- H.264 decode: 1280 × 720, 30 fps, Baseline, Main, and High Profile
- H.264 encode: D1, Baseline Profile
- MPEG-2 decode: 1280 × 720, 30 fps, MP-ML
- MPEG-2 encode: D1, MP-ML (in Software with Partial Acceleration in Hardware)
- VC-1 decode: 1280 × 720, 30 fps, Simple, Main, and Advanced Profile
- DivX decode: 1280 × 720, 30 fps Versions 3, 4, and 5
- RV10 decode: 1280 × 720, 30 fps
- MJPEG decode: 32 MPix/s
- MJPEG encode: 64 Mpix/s

— Graphics Processing Unit (GPU), 3D graphics processing compliant with the following:
- OpenGL ES Common Profile v1.0
- OpenGL ES Common Profile v1.1/Direct3D Mobile
- OpenGL ES Profile v2.0
- OpenVG 1.0

— Graphics Processing Unit (GPU2D), 2D graphics processing

— Audio
- Audio codecs are provided by software, which runs on ARM core
- 3 × SSIs
- Audio Mux
- SPDIF

- Security is supported by:
  - — High Assurance Boot (HAB) System
  - — ARM TrustZone (TZ) Trusted Execution environment
  - — IC Identification Module (IIM) and Central Security Unit (CSU)
  - — On-chip One-Time programmable electrical fuse array (E-Fusebox)
  - — RTIC: Real-Time Integrity Checker
  - — SAHARA Lite (SAHARA-Lite) cryptographic acceleration engine
  - — SJC—Secured JTAG controller
  - — Secure Real Time Clock (SRTC)
  - — Security Controller (SCC) with secure RAM
  - — Tamper Detection
  - — TrustZone Watchdog (TZ WDOG)
- Connectivity peripherals, timers, and External Memory Interface (EMI)
  - — Low-level communication protocols
  - — Embedded DMAs
  - — 3.3-V IO voltage for seamless integration
  - — USB 2.0 with integrated PHY/Transceiver, Serial FS (requires external transceiver)

— Video encoder with NTSC/PAL output
— DDR, Nand Flash (MLC 4/8-bit ECC) memory interface via EMI
— Timers: 2 × EPIT, GPT, and Watch Dog timer (WDOG)
— Miscellaneous connectivity support—I2C, SPI, UART, PWM, and Keypad interface

1.4 i.MX51 Modules List

Table 1 lists the modules used by the various subsystems of the i.MX51.

Table 1. i.MX51 Digital and Analog Modules

Block Mnemonic	Block Name	Subsystem	Brief Description
1-WIRE	1-Wire Interface	Connectivity Peripherals	1-Wire support provided for interfacing with an on-board EEPROM, and smart battery interfaces, for example: Dallas DS2502.
ARM Cortex A8™	ARM Cortex A8™ Platform	ARM	The ARM Cortex A8™ Core Platform consists of the ARM Cortex A8™ processor version r2p5 (with TrustZone) and its essential sub-blocks. It contains the Level 2 Cache Controller, 32-Kbyte L1 instruction cache, 32-Kbyte L1 data cache, and a 256-Kbyte L2 cache. The platform also contains an Event Monitor and Debug modules. It also has a NEON co-processor with SIMD media processing architecture, register file with 32 × 64-bit general-purpose registers, an Integer execute pipeline (ALU, Shift, MAC), dual, single-precision floating point execute pipeline (FADD, FMUL), load/store and permute pipeline and a Non-Pipelined Vector Floating Point (VFP) co-processor (VFPv3).
Audio Subsystem	Audio Subsystem	Multimedia Peripherals	The elements of the audio subsystem are three Synchronous Serial Interfaces (SSI1-3), a Digital Audio Mux (AUDMUX), and Digital Audio Out (SPDIF TX). See the specific interface listings in this table.
AUDMUX	Digital Audio Mux	Multimedia Peripherals	The AUDMUX is a programmable interconnect for voice, audio, and synchronous data routing between host serial interfaces (for example, SSI1, SSI2, and SSI3) and peripheral serial interfaces (audio and voice codecs). The AUDMUX has seven ports (three internal and four external) with identical functionality and programming models. A desired connectivity is achieved by configuring two or more AUDMUX ports.
CCM GPC SRC	Clock Control Module Global Power Controller System Reset Controller	Clocks, Resets, and Power Control	These modules are responsible for clock and reset distribution in the system, and also for system power management. The modules include three PLLs and a Frequency Pre-Multiplier (FPM).
CSPI-1, eCSPI-2 eCSPI-3	Configurable SPI, Enhanced CSPI	Connectivity Peripherals	Full-duplex enhanced Synchronous Serial Interface, with data rate up to 66.5Mbit/s (for eCSPI, master mode). It is configurable to support Master/Slave modes, four chip selects to support multiple peripherals.
CSU	Central Security Unit	Security	The Central Security Unit (CSU) is responsible for setting comprehensive security policy within the i.MX51 platform, and for sharing security information between the various security modules. The Security Control Registers (SCR) of the CSU are set during boot time by the High Assurance Boot (HAB) code and are locked to prevent further writing.

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
Debug System	Debug System	System Control	The Debug System provides real-time trace debug capability of both instructions and data. It supports a trace protocol that is an integral part of the ARM Real Time Debug solution (RealView). Real-time tracing is controlled by specifying a set of triggering and filtering resources, which include address and data comparators, cross-system triggers, counters, and sequencers.
EMI	External Memory Interface	Connectivity Peripherals	The EMI is an external and internal memory interface. It performs arbitration between multi-AXI masters to multi-memory controllers, divided into four major channels: fast memories (Mobile DDR, DDR2) channel, slow memories (NOR-FLASH/PSRAM/NAND-FLASH etc.) channel, internal memory (RAM, ROM) channel and graphical memory (GMEM) Channel. In order to increase the bandwidth performance, the EMI separates the buffering and the arbitration between different channels so parallel accesses can occur. By separating the channels, slow accesses do not interfere with fast accesses. EMI features: • 64-bit and 32-bit AXI ports • Enhanced arbitration scheme for fast channel, including dynamic master priority, and taking into account which pages are open or closed and what type (Read or Write) was the last access • Flexible bank interleaving • Supports 16/32-bit Mobile DDR up to 200 MHz SDCLK (mDDR400) • Supports 16/32-bit (Non-Mobile) DDR2 up to 200 MHz SDCLK (DDR2-400) • Supports up to 2 Gbit Mobile DDR memories • Supports 16-bit (in muxed mode only) PSRAM memories (sync and async operating modes), at slow frequency, for debugging purposes • Supports 32-bit NOR-Flash memories (only in muxed mode), at slow frequencies for debugging purposes • Supports 4/8-ECC, page sizes of 512 Bytes, 2 KBytes and 4 KBytes • NAND-Flash (including MLC) • Multiple chip selects • Enhanced Mobile DDR memory controller, supporting access latency hiding • Supports watermarking for security (Internal and external memories) • Supports Samsung OneNAND™ (only in muxed I/O mode)
EPIT-1 EPIT-2	Enhanced Periodic Interrupt Timer	Timer Peripherals	Each EPIT is a 32-bit “set and forget” timer that starts counting after the EPIT is enabled by software. It is capable of providing precise interrupts at regular intervals with minimal processor intervention. It has a 12-bit prescaler for division of input clock frequency to get the required time setting for the interrupts to occur, and counter values can be programmed on the fly.

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
eSDHC-1 eSDHC-2 eSDHC-3	Enhanced Multi-Media Card/ Secure Digital Host Controller	Connectivity Peripherals	The features of the eSDHC module, when serving as host, include the following: • Conforms to SD Host Controller Standard Specification version 2.0 • Compatible with the MMC System Specification version 4.2 • Compatible with the SD Memory Card Specification version 2.0 • Compatible with the SDIO Card Specification version 1.2 • Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, MMC and MMC RS cards • Configurable to work in one of the following modes: —SD/SDIO 1-bit, 4-bit —MMC 1-bit, 4-bit, 8-bit • Full-/high-speed mode • Host clock frequency variable between 32 kHz to 52 MHz • Up to 200 Mbps data transfer for SD/SDIO cards using four parallel data lines • Up to 416 Mbps data transfer for MMC cards using eight parallel data lines
eSDHC-4 (muxed with P-ATA)	Enhanced Multi-Media Card/ Secure Digital Host Controller	Connectivity Peripherals	Can be configured as eSDHC (see above) and is muxed with the P-ATA interface.
FEC	Fast Ethernet Controller	Connectivity Peripherals	The Ethernet Media Access Controller (MAC) is designed to support both 10 Mbps and 100 Mbps ethernet/IEEE Std 802.3™ networks. An external transceiver interface and transceiver function are required to complete the interface to the media.
FIRI	Fast Infra-Red Interface	Connectivity Peripherals	Fast Infra-Red Interface
GPIO-1 GPIO-2 GPIO-3 GPIO-4	General Purpose I/O Modules	System Control Peripherals	These modules are used for general purpose input/output to external ICs. Each GPIO module supports up to 32 bits of I/O.
GPT	General Purpose Timer	Timer Peripherals	Each GPT is a 32-bit “free-running” or “set and forget” mode timer with a programmable prescaler and compare and capture register. A timer counter value can be captured using an external event, and can be configured to trigger a capture event on either the leading or trailing edges of an input pulse. When the timer is configured to operate in “set and forget” mode, it is capable of providing precise interrupts at regular intervals with minimal processor intervention. The counter has output compare logic to provide the status and interrupt at comparison. This timer can be configured to run either on an external clock or on an internal clock.
GPU	Graphics Processing Unit	Multimedia Peripherals	The GPU provides hardware acceleration for 2D and 3D graphics algorithms with sufficient processor power to run desk-top quality interactive graphics applications on displays up to HD720 resolution. It supports color representation up to 32 bits per pixel. The GPU with its 128 KByte memory enables high performance mobile 3D and 2D vector graphics at rates up to 27 Mtriangles/sec, 166 M pixels/sec, 664 Mpixels/sec (Z).

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
GPU2D	Graphics Processing Unit-2D Ver. 1	Multimedia Peripherals	The GPU2D provides hardware acceleration for 2D graphic algorithms with sufficient processor power to run desk-top quality interactive graphics applications on displays up to HD720 resolution.
I^2C-1 I^2C-2 HS-I^2C	I^2C Interface	Connectivity Peripherals	I^2C provides serial interface for controlling peripheral devices. Data rates of up to 400 Kbps are supported by two of the I^2C ports. Data rates of up to 3.4 Mbps (I^2C Specification v2.1) are supported by the HS-I^2C. **Note:** See the errata for the HS-I^2C in the i.MX51 Chip Errata. The two standard I^2C modules have no errata.
IIM	IC Identification Module	Security	The IC Identification Module (IIM) provides an interface for reading, programming, and/or overriding identification and control information stored in on-chip fuse elements. The module supports electrically programmable poly fuses (e-Fuses). The IIM also provides a set of volatile software-accessible signals that can be used for software control of hardware elements not requiring non-volatility. The IIM provides the primary user-visible mechanism for interfacing with on-chip fuse elements. Among the uses for the fuses are unique chip identifiers, mask revision numbers, cryptographic keys, JTAG secure mode, boot characteristics, and various control signals requiring permanent non-volatility. The IIM also provides up to 28 volatile control signals. The IIM consists of a master controller, a software fuse value shadow cache, and a set of registers to hold the values of signals visible outside the module.
IOMUXC	IOMUX Control	System Control Peripherals	This module enables flexible I/O multiplexing. Each I/O pad has default as well as several alternate functions. The alternate functions are software configurable.
IPU	Image Processing Unit	Multimedia Peripherals	IPU enables connectivity to displays and image sensors, relevant processing and synchronization. It supports two display ports and two camera ports, through the following interfaces. • Legacy Interfaces • Analog TV interfaces (through a TV encoder bridge) The processing includes: • Support for camera control • Image enhancement: color adjustment and gamut mapping, gamma correction and contrast enhancement, sharpening and noise reduction • Video/graphics combining • Support for display backlight reduction • Image conversion—resizing, rotation, inversion and color space conversion • Synchronization and control capabilities, allowing autonomous operation. • Hardware de-interlacing support
KPP	Keypad Port	Connectivity Peripherals	The KPP supports an 8×8 external keypad matrix. The KPP features are as follows: • Open drain design • Glitch suppression circuit design • Multiple keys detection • Standby key press detection

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
P-ATA (Muxed with eSDHC-4	Parallel ATA	Connectivity Peripherals	The P-ATA block is an AT attachment host interface. Its main use is to interface with hard disc drives and optical disc drives. It interfaces with the ATA-5 (UDMA-4) compliant device over a number of ATA signals. It is possible to connect a bus buffer between the host side and the device side. This is muxed with eSDHC-4 interfaces.
PWM-1 PWM-2	Pulse Width Modulation	Connectivity Peripherals	The pulse-width modulator (PWM) has a 16-bit counter and is optimized to generate sound from stored sample audio images. It can also generate tones. The PWM uses 16-bit resolution and a 4x16 data FIFO to generate sound.
RAM 128 Kbytes	Internal RAM	Internal Memory	Unified RAM, can be split between Secure RAM and Non-Secure RAM
ROM 36 Kbytes	Boot ROM	Internal Memory	Supports secure and regular Boot Modes
RTIC	Real Time Integrity Checker	Security	Protecting read-only data from modification is one of the basic elements in trusted platforms. The Run-Time Integrity Checker v3 (RTICv3) module, is a data monitoring device responsible for ensuring that memory content is not corrupted during program execution. The RTICv3 mechanism periodically checks the integrity of code or data sections during normal OS run-time execution without interfering with normal operation. The RTICv3's purpose is to ensure the integrity of the peripheral memory contents, protect against unauthorized external memory elements replacement, and assist with boot authentication.
SAHARA Lite	SAHARA security accelerator Lite	Security	SAHARA (Symmetric/Asymmetric Hashing and Random Accelerator) is a security co-processor. It implements symmetric encryption algorithms, (AES, DES, 3DES, and RC4), public key algorithms, hashing algorithms (MD5, SHA-1, SHA-224, and SHA-256), and a hardware random number generator. It has a slave IP bus interface for the host to write configuration and command information, and to read status information. It also has a DMA controller, with an AHB bus interface, to reduce the burden on the host to move the required data to and from memory.
SCC	Security Controller	Security	The Security Controller is a security assurance hardware module designed to safely hold sensitive data such as encryption keys, digital right management (DRM) keys, passwords, and biometrics reference data. The SCC monitors the system's alert signal to determine if the data paths to and from it are secure—that is, cannot be accessed from outside of the defined security perimeter. If not, it erases all sensitive data on its internal RAM. The SCC also features a Key Encryption Module (KEM) that allows non-volatile (external memory) storage of any sensitive data that is temporarily not in use. The KEM utilizes a device-specific hidden secret key and a symmetric cryptographic algorithm to transform the sensitive data into encrypted data.

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
SDMA	Smart Direct Memory Access	System Control Peripherals	The SDMA is multi-channel flexible DMA engine. It helps in maximizing system performance by off loading various cores in dynamic data routing. The SDMA features list is as follows: • Powered by a 16-bit instruction-set micro-RISC engine • Multi-channel DMA supports up to 32 time-division multiplexed DMA channels • 48 events with total flexibility to trigger any combination of channels • Memory accesses including linear, FIFO, and 2D addressing • Shared peripherals between ARM Cortex A8™ and SDMA • Very fast context-switching with two-level priority-based preemptive multi-tasking • DMA units with auto-flush and prefetch capability • Flexible address management for DMA transfers (increment, decrement, and no address changes on source and destination address) • DMA ports can handle unit-directional and bi-directional flows (copy mode) • Up to 8-word buffer for configurable burst transfers for EMI • Support of byte-swapping and CRC calculations • A library of scripts and API are available
SIM	Subscriber Identity Module Interface	Connectivity Peripherals	The SIM is an asynchronous interface with additional features for allowing communication with Smart Cards conforming to the ISO 7816 specification. The SIM is designed to facilitate communication to SIM cards or pre-paid phone cards.
SJC	Secure JTAG Interface	System Control Peripherals	JTAG manipulation is a known hacker's method of executing unauthorized program code, getting control over secure applications, and running code in privileged modes. The JTAG port provides a debug access to several hardware blocks including the ARM processor and the system bus. The JTAG port must be accessible during platform initial laboratory bring-up, manufacturing tests and troubleshooting, as well as for software debugging by authorized entities. However, in order to properly secure the system, unauthorized JTAG usage should be strictly forbidden. In order to prevent JTAG manipulation while allowing access for manufacturing tests and software debugging, the i.MX51 processor incorporates a mechanism for regulating JTAG access. The i.MX51Secure JTAG Controller provides four different JTAG security modes that can be selected via e-fuse configuration.
SPBA	Shared Peripheral Bus Arbiter	System Control Peripherals	SPBA (Shared Peripheral Bus Arbiter) is a two-to-one IP bus interface (IP bus) arbiter.
SPDIF	Sony Philips Digital Interface	Multimedia Peripherals	A standard digital audio transmission protocol developed jointly by the Sony and Philips corporations. Only the transmitter functionality is supported.
SRTC	Secure Real Time Clock	Security	The SRTC incorporates a special System State Retention Register (SSRR) that stores system parameters during system shutdown modes. This register and all SRTC counters are powered by dedicated supply rail NVCC_SRTC_POW. The NVCC_SRTC_POW can be energized even if all other supply rails are shut down. This register is helpful for storing warm boot parameters. The SSRR also stores the system security state. In case of a security violation, the SSRR mark the event (security violation indication).

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
SSI-1 SSI-2 SSI-3	I2S/SSI/AC97 Interface	Connectivity Peripherals	The SSI is a full-duplex synchronous interface used on the i.MX51 processor to provide connectivity with off-chip audio peripherals. The SSI supports a wide variety of protocols (SSI normal, SSI network, I2S, and AC-97), bit depths (up to 24 bits per word), and clock/frame sync options. Each SSI has two pairs of 8x24 FIFOs and hardware support for an external DMA controller in order to minimize its impact on system performance. The second pair of FIFOs provides hardware interleaving of a second audio stream, which reduces CPU overhead in use cases where two timeslots are being used simultaneously.
TVE	TV Encoder	Multimedia	The TVE is implemented in conjunction with the Image Processing Unit (IPU) allowing handheld devices to display captured still images and video directly on a TV or LCD projector. It supports the following analog video outputs: composite, S-video, and component video up to HD720p/1080i.
TZIC	TrustZone Aware Interrupt Controller	ARM/Control	The TrustZone Interrupt Controller (TZIC) collects interrupt requests from all i.MX51 sources and routes them to the ARM core. Each interrupt can be configured as a normal or a secure interrupt. Software Force Registers and software Priority Masking are also supported.
UART-1 UART-2 UART-3	UART Interface	Connectivity Peripherals	Each of the UART modules supports the following serial data transmit/receive protocols and configurations: • 7 or 8 bit data words, 1 or 2 stop bits, programmable parity (even, odd, or none) • Programmable baud rates up to 4 MHz. This is a higher max baud rate relative to the 1.875 MHz, which is stated by the TIA/EIA-232-F standard and previous Freescale UART modules. • 32-byte FIFO on Tx and 32 half-word FIFO on Rx supporting auto-baud • IrDA 1.0 support (up to SIR speed of 115200 bps) • Option to operate as 8-pins full UART, DCE, or DTE
USB	USB 2.0 High-Speed OTG and 3x Hosts	Connectivity Peripherals	USB-OTG contains one high-speed OTG module, which is internally connected to the on-chip HS USB PHY. There are an additional three high-speed host modules that require external USB PHYs.
VPU	Video Processing Unit	Multimedia Peripherals	A high-performing video processing unit (VPU), which covers many SD-level video decoders and SD-level encoders as a multi-standard video codec engine as well as several important video processing such as rotation and mirroring. VPU Features: • MPEG-4 decode: 720p, 30 fps, simple profile and advanced simple profile • MPEG-4 encode: D1, 25/30 fps, simple profile • H.263 decode: 720p, 30 fps, profile 3 • H.263 encode: D1, 25/30 fps, profile 3 • H.264 decode: 720p, 30 fps, baseline, main, and high profile • H.264 encode: D1, 25/30 fps, baseline profile • MPEG-2 decode: 720p, 30 fps, MP-ML • MPEG-2 encode: D1, 25/30 fps, MP-ML (in software with partial acceleration in hardware) • VC-1 decode: 720p, 30 fps, simple, main, and advanced profile • DivX decode: 720p, 30 fps versions 3, 4, and 5 • RV10 decode: 720p, 30 fps • MJPEG decode: 32 Mpix/s • MJPEG encode: 64 Mpix/s

Table 1. i.MX51 Digital and Analog Modules (continued)

Block Mnemonic	Block Name	Subsystem	Brief Description
WDOG-1	Watch Dog	Timer Peripherals	The Watch Dog Timer supports two comparison points during each counting period. Each of the comparison points is configurable to evoke an interrupt to the ARM core, and a second point evokes an external event on the WDOG line.
WDOG-2 (TZ)	Watch Dog (TrustZone)	Timer Peripherals	The TrustZone Watchdog (TZ WDOG) timer module protects against TrustZone starvation by providing a method of escaping normal mode and forcing a switch to the TZ mode. TZ starvation is a situation where the normal OS prevents switching to the TZ mode. This situation should be avoided, as it can compromise the system's security. Once the TZ WDOG module is activated, it must be serviced by TZ software on a periodic basis. If servicing does not take place, the timer times out. Upon a time-out, the TZ WDOG asserts a TZ mapped interrupt that forces switching to the TZ mode. If it is still not served, the TZ WDOG asserts a security violation signal to the CSU. The TZ WDOG module cannot be programmed or deactivated by a normal mode SW.
XTALOSC	Crystal Oscillator I/F	Clocking	The XTALOSC module allows connectivity to an external crystal.

1.5 Frequency Requirements

The i.MX51 datasheet lists the target system frequencies at different voltage conditions.

1.6 Memory Interfaces

The EMI supports the following memory interfaces:

- mDDR, DDR2, 32-bit, 200 MHz clock
- NAND (MLC/SLC) Flash, 8/16-bit, 40 MHz
- OneNAND, 1-2 Kbyte page size

Chapter 2
Memory Map

This chapter introduces the memory architecture of the i.MX51. The system memory high-level partition is defined below:

- Level 1 Cache
  - Instruction (32-Kbyte)
  - Data (32-Kbyte)
- Level 2 Cache
  - Unified instruction and data (256 Kbytes)
- Level 2 Memory
  - Boot ROM, including HAB (36-Kbyte)
  - Unified Internal RAM (128 Kbyte), can be split between Secure and Non-Secure RAM
- External memory interface
  - 32-bit mDDR, up to 200 MHz clock
  - 32-bit DDR2, up to 200 MHz clock
  - 8/16-bit NAND SLC/MLC Flash, up to 40 MHz
  - 8/16-bit NOR Flash.

2.1 CPU Memory Map

Table 2-1 shows the system memory map.

NOTE

User must not address reserved memory regions. Access to reserved memory regions can cause unpredictable behavior.

Table 2-1. i.MX51 System Memory Map

AP		Size	Region
Start Address	End Address		
On-Chip Memories (hardware connection via EMI)			
0000_0000	0000_8FFF	36K	Boot ROM
0000_9000	1FFD_FFFF	512M (minus 164K)	Reserved for Internal ROM aliasing
1FFE_0000	1FFF_FFFF	128K	SCC RAM
2000_0000	2001_FFFF	128K	Graphics Memory of GPU

Table 2-1. i.MX51 System Memory Map (continued)

AP		Size	Region
Start Address	**End Address**		
2002_0000	2FFF_FFFF	256M (minus 128K)	Reserved
On Chip AHB Accessed IPs			
3000_0000	3FFF_FFFF	256M	GPU
4000_0000	5FFF_FFFF	512M	IPUEX
On Chip AHB Accessed IPs—Debug APB			
6000_0000	6000_0FFF	4K	Debug ROM
6000_1000	6000_1FFF	4K	ETB
6000_2000	6000_2FFF	4K	ETM
6000_3000	6000_3FFF	4K	TPIU
6000_4000	6000_4FFF	4K	CTI0
6000_5000	6000_5FFF	4K	CTI1
6000_6000	6000_6FFF	4K	CTI2
6000_7000	6000_7FFF	4K	CTI3
6000_8000	6000_8FFF	4K	Cortex Debug Unit
6000_9000	6FFF_FFFF	256M (minus 36K)	Reserved
AIPS_TZ#1			
AIPS_TZ#1- SPBA IPs, Mapped to global module enable 0			
7000_0000	7000_3FFF	16K	Reserved
7000_4000	7000_7FFF	16K	ESDHC 1
7000_8000	7000_BFFF	16K	ESDHC 2
7000_C000	7000_FFFF	16K	UART 3
7001_0000	7001_3FFF	16K	eCSPI1
7001_4000	7001_7FFF	16K	SSI2
7001_8000	7001_BFFF	16K	Reserved
7001_C000	7001_FFFF	16K	Reserved for SDMA internal registers
7002_0000	7002_3FFF	16K	ESDHC 3
7002_4000	7002_7FFF	16K	ESDHC 4
7002_8000	7002_BFFF	16K	SPDIF
7002_C000	7002_FFFF	16K	Reserved
7003_0000	7003_3FFF	16K	PATA (PORT UDMA)

Table 2-1. i.MX51 System Memory Map (continued)

AP		Size	Region
Start Address	**End Address**		
7003_4000	7003_7FFF	16K	SLM
7003_8000	7003_BFFF	16K	HSI2C
7003_C000	7003_FFFF	16K	SPBA
AIPS_TZ#1- Global Module Enables			
7004_0000	71FF_FFFF	32M (minus 256K)	Reserved AIPS_TZ #1 off platform global module enable #0
7200_0000	73EF_FFFF	31M	Reserved AIPS_TZ #1 off platform global module enable #1
AIPS_TZ#1- On Platform			
73F0_0000	73F7_FFFF	512K	Reserved AIPS_TZ #1 on platform slots
AIPS_TZ#1- Off Platform			
73F8_0000	73F8_3FFF	16K	USBOH3 (PORT USB)
73F8_4000	73F8_7FFF	16K	GPIO1
73F8_8000	73F8_BFFF	16K	GPIO2
73F8_C000	73F8_FFFF	16K	GPIO3
73F9_0000	73F9_3FFF	16K	GPIO4
73F9_4000	73F9_7FFF	16K	KPP
73F9_8000	73F9_BFFF	16K	WDOG1
73F9_C000	73F9_FFFF	16K	WDOG2 (TZ)
73FA_0000	73FA_3FFF	16K	GPT
73FA_4000	73FA_7FFF	16K	SRTC
73FA_8000	73FA_BFFF	16K	IOMUXC
73FA_C000	73FA_FFFF	16K	EPIT1
73FB_0000	73FB_3FFF	16K	EPIT2
73FB_4000	73FB_7FFF	16K	PWM1
73FB_8000	73FB_BFFF	16K	PWM2
73FB_C000	73FB_FFFF	16K	UART 1
73FC_0000	73FC_3FFF	16K	UART 2
73FC_4000	73FC_7FFF	16K	USBOH3 (PORT PL301)
73FC_8000	73FC_BFFF	16K	Reserved
73FC_C000	73FC_FFFF	16K	Reserved
73FD_0000	73FD_3FFF	16K	SRC

Table 2-1. i.MX51 System Memory Map (continued)

AP		Size	Region
Start Address	**End Address**		
73FD_4000	73FD_7FFF	16K	CCM
73FD_8000	73FD_BFFF	16K	GPC
73FD_C000	73FD_FFFF	16K	Reserved
73FE_0000	73FE_3FFF	16K	Reserved
73FE_4000	73FE_7FFF	16K	Reserved
73FE_8000	73FE_BFFF	16K	Reserved
73FE_C000	73FE_FFFF	16K	Reserved
73FF_0000	73FF_3FFF	16K	Reserved
73FF_4000	73FF_FFFF	48K	Reserved AIPS_TZ #1 off platform space.
7400_0000	7FFF_FFFF	448M	Reserved (Aliased to AIPS_TZ#1 slots)
AIPS_TZ#2- Global Module Enables			
8000_0000	81FF_FFFF	32M	Reserved AIPS_TZ #1 off platform global module enable #0
8200_0000	83EF_FFFF	31M	Reserved AIPS_TZ #1 off platform global module enable #1
AIPS_TZ#2- On Platform			
83F0_0000	83F7_FFFF	512K	Reserved AIPS_TZ #2 on platform slots
AIPS_TZ#2- Off Platform			
83F8_0000	83F8_3FFF	16K	DPLLIP1
83F8_4000	83F8_7FFF	16K	DPLLIP2
83F8_8000	83F8_BFFF	16K	DPLLIP3
83F8_C000	83F8_FFFF	16K	Reserved
83F9_0000	83F9_3FFF	16K	Reserved
83F9_4000	83F9_7FFF	16K	AHBMAX
83F9_8000	83F9_BFFF	16K	IIM
83F9_C000	83F9_FFFF	16K	CSU
83FA_0000	83FA_3FFF	16K	TIGERP_PLATFORM_NE_32K_256K
83FA_4000	83FA_7FFF	16K	OWIRE
83FA_8000	83FA_BFFF	16K	FIRI
83FA_C000	83FA_FFFF	16K	eCSPI2
83FB_0000	83FB_3FFF	16K	SDMA (port IPS_HOST)
83FB_4000	83FB_7FFF	16K	SCC

Table 2-1. i.MX51 System Memory Map (continued)

AP		Size	Region
Start Address	**End Address**		
83FB_8000	83FB_BFFF	16K	ROMCP
83FB_C000	83FB_FFFF	16K	RTIC
83FC_0000	83FC_3FFF	16K	CSPI
83FC_4000	83FC_7FFF	16K	I2C2
83FC_8000	83FC_BFFF	16K	I2C1
83FC_C000	83FC_FFFF	16K	SSI1
83FD_0000	83FD_3FFF	16K	AUDMUX
83FD_4000	83FD_7FFF	16K	Reserved
83FD_8000	83FD_BFFF	16K	EMI[1]
83FE_0000	83FE_3FFF	16K	PATA (PORT PIO)
83FE_4000	83FE_7FFF	16K	SIM
83FE_8000	83FE_BFFF	16K	SSI3
83FE_C000	83FE_FFFF	16K	FEC
83FF_0000	83FF_3FFF	16K	TVE
83FF_4000	83FF_7FFF	16K	VPU
83FF_8000	83FF_BFFF	16K	SAHARA Lite
83FF_C000	83FF_FFFF	16K	Reserved
8400_0000	8FFF_BFFF	448M (minus 256K)	Reserved (aliased AIPS_TZ #2 slots)
On Chip AHB Accessed IPs			
8FFF_C000	8FFF_FFFF	16K	Reserved
Off Chip Memories (HW connection via EMI)			
9000_0000	9FFF_FFFF	256M	CSD0 DDR
A000_0000	AFFF_FFFF	256M	CSD1 DDR
B000_0000	B7FF_FFFF	128M	CS0 (Flash) 128M
B800_0000	BFFF_FFFF	128M	CS1 (Flash) 128M
C000_0000	C7FF_FFFF	128M	CS2 (Flash) 128M
C800_0000	CBFF_FFFF	64M	CS3 (Flash) 64MB
CC00_0000	CDFF_FFFF	32M	CS4 (SRAM) 32MB
CE00_0000	CFFE_FFFF	32M (minus 64K)	CS5 (SRAM) 32MB
CFFF_0000	CFFF_FFFF	64K	NAND FLASH (internal buffer)[2]
On Chip AHB Accessed IPs			

Table 2-1. i.MX51 System Memory Map (continued)

AP		Size	Region
Start Address	End Address		
D000_0000	DFFF_FFFF	256M	GPU2D (OpenVG)
E000_0000	E000_3FFF	16K	TZIC
ED00_4000	FFFF_FFFF	512M (minus 16K)	Reserved

1 The memory map of M4IF, ESDCTL, WEIM, and NFC is specified in the Section 27.4.1, External/Internal Memory Map, of Chapter 27, "External Memory Interface (EMI)."

2 Also named as AXI_BASE in Section 45.6.1, Memory Map of Chapter 45, "NAND Flash Controller (NFC)."

2.2 DMA Memory Map

The Smart DMA's memory map is defined in Table 2-2.

NOTE

User must not address reserved memory regions. Access to reserved memory regions can produce unpredictable behavior.

Table 2-2. SDMA Peripheral Memory Map

Peripheral	Base Address	Size	Comments
Reserved for SDMA internal memory	0x0000	4KB	—
ESDHC-1	0x1000	4KB	—
ESDHC-2	0x2000	4KB	—
UART 3	0x3000	4KB	—
eCSPI1	0x4000	4KB	—
SSI2	0x5000	4KB	—
Reserved	0x6000	4KB	Reserved
Reserved for SDMA internal registers	0x7000	4KB	—
ESDHC-3 (CE-ATA)	0x8000	4KB	—
ESDHC-4	0x9000	4KB	—
SPDIF	0xA000	4KB	—
Reserved	0xB000	4KB	Reserved
PATA	0xC000	4KB	—
Reserved	0xD000	4KB	Reserved
HS-I2C	0xE000	4KB	—
SPBA Registers	0xF000	4KB	—

Chapter 3
Interrupts and DMA Events

3.1 Overview

This chapter provides information about the assignments of interrupts in Section 3.2, AP Interrupts," and about the DMA events in Section 3.3, SDMA Event Mapping."

3.2 AP Interrupts

The TrustZone interrupt controller (TZIC) collects up to 128 interrupt requests from all i.MX51 sources and provides an interface to the core. Each interrupt can be configured as a normal or a secure interrupt. Software force registers and software priority masking are also supported.

Table 3-2 describes the ARM interrupt sources.

Table 3-2. ARM Domain Interrupt Summary

IRQ	Interrupt Source	Interrupt Description
0	Reserved	Reserved
1	eSDHC1	Enhanced SDHC Interrupt Request
2	eSDHC2	Enhanced SDHC Interrupt Request
3	eSDHC3	CE-ATA Interrupt Request based on eSDHC-3
4	eSDHC4	Enhanced SDHC Interrupt Request
5	DAP	Power-up Request
6	SDMA	"AND" of all 48 interrupts from all the channels
7	IOMUX	POWER FAIL interrupt
8	EMI (NFC)	nfc interrupt out
9	VPU	VPU Interrupt Request
10	IPUEX	IPUEX Error Interrupt
11	IPUEX	IPUEX Sync Interrupt
12	GPU3D	GPU3D Interrupt Request
13	Reserved	Reserved
14	USBOH3	USB Host 1
15	EMI	Consolidated EMI Interrupt

Table 3-2. ARM Domain Interrupt Summary (continued)

IRQ	Interrupt Source	Interrupt Description
16	USBOH3	USB Host 2
17	USBOH3	USB Host 3
18	USBOH3	USB OTG
19	SAHARA Lite	SAHARA host 0 (TrustZone) Intr
20	SAHARA Lite	SAHARA host 1 (non-TrustZone) Intr
21	SCC	Security Monitor High Priority Interrupt Request.
22	SCC	Secure (TrustZone) Interrupt Request.
23	SCC	Regular (Non-Secure) Interrupt Request.
24	SRTC	SRTC Consolidated Interrupt. Non TZ.
25	SRTC	SRTC Security Interrupt. TZ.
26	RTIC	RTIC (Trust Zone) Interrupt Request. Indicates that the RTIC has completed hashing the selected memory block(s) during single-hash/boot mode.
27	CSU	CSU Interrupt Request 1. Indicates to the processor that one or more alarm inputs were asserted
28	Reserved	Reserved
29	SSI1	SSI-1 Interrupt Request
30	SSI2	SSI-2 Interrupt Request
31	UART 1	UART-1 ORed interrupt
32	UART 2	UART-2 ORed interrupt
33	UART 3	UART-3 ORed interrupt
34	Reserved	Reserved
35	Reserved	Reserved
36	eCSPI1	eCSPI1 interrupt request line to the core.
37	eCSPI2	eCSPI2 interrupt request line to the core.
38	CSPI	CSPI interrupt request line to the core.
39	GPT	“OR” of GPT Rollover interrupt line, Input Capture 1 and 2 lines, Output Compare 1,2, and 3 Interrupt lines
40	EPIT1	EPIT1 output compare interrupt
41	EPIT2	EPIT2 output compare interrupt
42	GPIO1	Active HIGH Interrupt from INT7 from GPIO
43	GPIO1	Active HIGH Interrupt from INT6 from GPIO
44	GPIO1	Active HIGH Interrupt from INT5 from GPIO

Table 3-2. ARM Domain Interrupt Summary (continued)

IRQ	Interrupt Source	Interrupt Description
45	GPIO1	Active HIGH Interrupt from INT4 from GPIO
46	GPIO1	Active HIGH Interrupt from INT3 from GPIO
47	GPIO1	Active HIGH Interrupt from INT2 from GPIO
48	GPIO1	Active HIGH Interrupt from INT1 from GPIO
49	GPIO1	Active HIGH Interrupt from INT0 from GPIO
50	GPIO1	Combined interrupt indication for GPIO1 signal 0 throughout 15
51	GPIO1	Combined interrupt indication for GPIO1 signal 16 throughout 31
52	GPIO2	Combined interrupt indication for GPIO2 signal 0 throughout 15
53	GPIO2	Combined interrupt indication for GPIO2 signal 16 throughout 31
54	GPIO3	Combined interrupt indication for GPIO3 signal 0 throughout 15
55	GPIO3	Combined interrupt indication for GPIO3 signal 16 throughout 31
56	GPIO4	Combined interrupt indication for GPIO4 signal 0 throughout 15
57	GPIO4	Combined interrupt indication for GPIO4 signal 16 throughout 31
58	WDOG1	Watchdog Timer reset
59	WDOG2	Watchdog Timer reset
60	KPP	Keypad Interrupt
61	PWM 1	Cumulative interrupt line.“OR” of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line.
62	I2C1	I2C-1 Interrupt
63	I2C2	I2C-2 Interrupt
64	HS-I2C	High Speed I2C Interrupt
65	Reserved	Reserved
66	Reserved	Reserved
67	SIM	SIM interrupt composed of oef, xte, sdi1, and sdi0
68	SIM	SIM interrupt composed of tc, etc, tfe, and rdrf
69	IIM	Interrupt request to the processor. Indicates to the processor that program or explicit sense cycle is completed successfully or in case of error. This signal is low-asserted.
70	PATA	Parallel ATA host controller interrupt request
71	CCM	CCM, Interrupt Request 1
72	CCM	CCM, Interrupt Request 2
73	GPC	GPC, Interrupt Request 1
74	GPC	GPC, Interrupt Request 2
75	SRC	SRC interrupt request

Table 3-2. ARM Domain Interrupt Summary (continued)

IRQ	Interrupt Source	Interrupt Description
76	TIGERP_PLATFORM_NE_32K_256K	Neon Monitor Interrupt
77	TIGERP_PLATFORM_NE_32K_256K	Performance Unit Interrupt
78	TIGERP_PLATFORM_NE_32K_256K	CTI IRQ
79	TIGERP_PLATFORM_NE_32K_256K	Debug Interrupt, from Cross-Trigger Interface 1
80	TIGERP_PLATFORM_NE_32K_256K	Debug Interrupt, from Cross-Trigger Interface 1
84	GPU2D	GPU2D (OpenVG) general interrupt
85	GPU2D	GPU2D (OpenVG) busy signal (for S/W power gating feasibility)
86	Reserved	Reserved
87	FEC	Fast Ethernet Controller Interrupt request (OR of 13 interrupt sources)
88	OWIRE	1-Wire Interrupt Request
89	TIGERP_PLATFORM_NE_32K_256K	Debug Interrupt, from Cross-Trigger Interface 2
90	SJC	—
91	SPDIF	—
92	TVE	—
93	FIRI	FIRI Intr (OR of all 4 interrupt sources)
94	PWM 2	Cumulative interrupt line.“OR” of Rollover Interrupt line, Compare Interrupt line and FIFO Waterlevel crossing interrupt line.
95	Reserved	Reserved
96	SSI3	SSI-3 Interrupt Request
97	EMI	Boot sequence completed interrupt
98	TIGERP_PLATFORM_NE_32K_256K	Debug Interrupt, from Cross-Trigger Interface 3
99	Reserved	Reserved
100	VPU	Idle interrupt from VPU (for S/W power gating)
101	EMI	Indicates all pages have been transferred to NFC during an auto program operation.
102	GPU3D	Idle interrupt from GPU3D (for S/W power gating)
103_128	Reserved	Reserved

3.3 SDMA Event Mapping

Table 3-3 shows the DMA request signals for peripherals.

Table 3-3. SDMA Event Mapping

Event Number	DMA Source	Description
0	VPU	VPU DMA request
1	GPC	Will be used for power management.
2	PATA	Rx FIFO of PATA
3	PATA	Tx FIFO of PATA
4	PATA	PATA Transfer End
5	Reserved	Reserved
6	eCSPI1	DMA Rx request
7	eCSPI1	DMA Tx request
8	eCSPI2	DMA Rx request
9	eCSPI2	DMA Tx request
10	HS-I^{2}C	HS I^{2}C DMA Tx request
11	HS-I^{2}C	HS I^{2}C DMA Rx request
12	FIRI	DMA request of receiver FIFO
13	FIRI	DMA request of transmitter FIFO
14	IOMUX	External DMA request from BGA contact GPIO1_4
15	IOMUX	External DMA request from BGA contact GPIO1_5
16	UART2	Rx FIFO of UART 2
17	UART2	Tx FIFO of UART 2
18	UART1	Rx FIFO of UART 1
19	UART1	Tx FIFO of UART 1
20	esdhc1 I^{2}C1	MMC/SDHC1 **muxed with** I^{2}C1
21	esdhc2 I^{2}C2	MMC/SDHC2 **muxed with** I^{2}C2
22	SSI2	SSI #2 receive 2 DMA request
23	SSI2 SLM	SSI #2 transmit 2 DMA request
24	SSI2 SLM	SSI #2 receive 1 DMA request
25	SSI2 SLM	SSI #2 transmit 1 DMA request
26	SSI1	SSI #1 receive 2 DMA request
27	SSI1	SSI #1 transmit 2 DMA request
28	SSI1	SSI #1 receive 1 DMA request
29	SSI1	SSI #1 transmit 1 DMA request

Table 3-3. SDMA Event Mapping (continued)

Event Number	DMA Source	Description
30	EMI	Asserts every time NFC finishes reading a page
31	CTI2	CTI2 (SDMA_CTI) trigger_out[0] connected to SDMA event.
32	EMI	Asserts at the beginning of auto-program sequence, and every time the NFC finishes transferring data from the RAM to the NAND (Meaning, the SDMA can write to the RAM the next page).
33	CTI2	CTI2 (SDMA_CTI) trigger_out[1] connected to SDMA event.
34	EPIT2	EPIT2 DMA request
35	SSI3 SLM	SSI #3 receive 2 DMA request
36	IPUEX	IPUEX DMA request
37	SSI3 SLM	SSI #3 transmit 2 DMA request
38	CSPI	DMA Rx request
39	CSPI	DMA Tx request
40	eSDHC3	MMC/SDHC3 DMA request
41	eSDHC4	MMC/SDHC4 DMA request
42	Reserved	Reserved
43	UART3	Rx FIFO of UART 3
44	UART3	Tx FIFO of UART 3
45	SPDIF	SPDIF DMA request
46	SSI3 SLM	SSI #3 receive 1 DMA request
47	SSI3 SLM	SSI #3 transmit 1 DMA request

As shown in the table, some of the events are an output of a mux of two signals or triggers. The select of this mux is controlled by general purpose register 0 in IOMUXC.

For other shared connectivity peripherals that do not have dedicated DMA request signals, the AP interrupt service routines have the option of programming the SDMA to move data between the peripheral and memory.

Chapter 4
External Signals and Pin Multiplexing

Because the IC has a limited number of pins, most pins have multiple signal options. The input-output multiplexer (IOMUX) controls the pin multiplexing. The IOMUX also configures other pin characteristics, such as voltage level, drive strength, and hysteresis.

4.1 External Signals

Table 4-1 shows the external signals of the i.MX51. Table 4-2 shows an additional view of external signals muxing. It presents the muxing options per module/instance. Table 4-3 shows IOMUX daisy chain settings.

4.1.1 I/O Configuration Parameters

In the Pad Settings column, each I/O configuration parameter uses one of the two following notations:

- A CFG (value) notation, where ‘value’ is High, Disabled, Keep, and others. The CFG notation indicates the parameter is configurable using the SW_PAD_CTL_PAD and SW_PAD_CTL_GRP registers of Appendix A, “IOMUX Controller (IOMUXC).” The value shown is the default setting, and the associated bits are shown in Appendix A, “IOMUX Controller (IOMUXC).”
- A value notation (with no CFG indicator) where ‘value’ is High, Disabled, Keep, and others indicates the parameter is not configurable. The value shown is the default setting. However, the associated bits shown in Appendix A, “IOMUX Controller (IOMUXC),” below the gray bit-location boxes (read-only) are simply the values read from that particular location and do not reflect the actual functional default.

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations

Pad Name	Mode	Instance	Port	Pad Settings
EIM_DA0	ALT0	emi	EIM_DA[0]	Drive Strength—CFG (High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/ Down Config.—100 KΩ PU Strength mode—4_level dse test—Regular Open Drain Enable—Disabled Pull/Keep Enable—CFG (Enabled) Slew Rate—CFG (FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[16]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_DA1	ALT0	emi	EIM_DA[1]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[17]	
EIM_DA2	ALT0	emi	EIM_DA[2]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[18]	
EIM_DA3	ALT0	emi	EIM_DA[3]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[19]	
EIM_DA4	ALT0	emi	EIM_DA[4]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[20]	
EIM_DA5	ALT0	emi	EIM_DA[5]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[21]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_DA6	ALT0	emi	EIM_DA[6]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[22]	
EIM_DA7	ALT0	emi	EIM_DA[7]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[23]	
EIM_DA8	ALT0	emi	EIM_DA[8]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[24]	
EIM_DA9	ALT0	emi	EIM_DA[9]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[25]	
EIM_DA10	ALT0	emi	EIM_DA[10]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[26]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_DA11	ALT0	emi	EIM_DA[11]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[27]	
EIM_DA12	ALT0	emi	EIM_DA[12]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[28]	
EIM_DA13	ALT0	emi	EIM_DA[13]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[29]	
EIM_DA14	ALT0	emi	EIM_DA[14]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[30]	
EIM_DA15	ALT0	emi	EIM_DA[15]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tpiu	TRACE[31]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_D16	ALT0	emi	WEIM_D[16]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[0]	
	ALT2	usboh3	USBH2_DATA0	
	ALT3	uart2	CTS	
	ALT4	i2c1	SDA	
	ALT5	audmux	AUD4_RXFS	
	ALT6	tpiu	TRACE[0]	
	ALT7	audmux	AUD5_TXD	
EIM_D17	ALT0	emi	WEIM_D[17]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[1]	
	ALT2	usboh3	USBH2_DATA1	
	ALT3	uart2	RXD_MUX	
	ALT4	uart3	CTS	
	ALT5	ipu	SISG[4]	
	ALT6	tpiu	TRACE[1]	
	ALT7	audmux	AUD5_RXD	
EIM_D18	ALT0	emi	WEIM_D[18]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[2]	
	ALT2	usboh3	USBH2_DATA2	
	ALT3	uart2	TXD_MUX	
	ALT4	uart3	RTS	
	ALT5	ipu	SISG[5]	
	ALT6	tpiu	TRACE[2]	
	ALT7	audmux	AUD5_TXC	
EIM_D19	ALT0	emi	WEIM_D[19]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[3]	
	ALT2	usboh3	USBH2_DATA3	
	ALT3	uart2	RTS	
	ALT4	i2c1	SCL	
	ALT5	audmux	AUD4_RXC	
	ALT6	tpiu	TRACE[3]	
	ALT7	audmux	AUD5_TXFS	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_D20	ALT0	emi	WEIM_D[20]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[4]	
	ALT2	usboh3	USBH2_DATA4	
	ALT3	csu	CSU_INT_DEB	
	ALT4	srtc	SRTC_ALARM_DEB	
	ALT5	audmux	AUD4_TXD	
	ALT6	tpiu	TRACE[4]	
	ALT7	usbphy	TXREADY	
EIM_D21	ALT0	emi	WEIM_D[21]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[5]	
	ALT2	usboh3	USBH2_DATA5	
	ALT3	srtc	SRTC_ALARM_DEB	
	ALT5	audmux	AUD4_RXD	
	ALT6	tpiu	TRACE[5]	
	ALT7	usbphy	RXVALID	
EIM_D22	ALT0	emi	WEIM_D[22]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[6]	
	ALT2	usboh3	USBH2_DATA6	
	ALT3	scc	FAIL_STATE	
	ALT5	audmux	AUD4_TXC	
	ALT6	tpiu	TRACE[6]	
	ALT7	usbphy	RXACTIVE	
EIM_D23	ALT0	emi	WEIM_D[23]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[7]	
	ALT2	usboh3	USBH2_DATA7	
	ALT3	scc	SEC_STATE	
	ALT4	spdif	OUT1	
	ALT5	audmux	AUD4_TXFS	
	ALT6	tpiu	TRACE[7]	
	ALT7	usbphy	RXERROR	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_D24	ALT0	emi	WEIM_D[24]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[8]	
	ALT2	usboh3	USBOTG_DATA0	
	ALT3	uart3	CTS	
	ALT4	i2c2	SDA	
	ALT5	audmux	AUD6_RXFS	
	ALT6	tpiu	TRACE[8]	
	ALT7	usbphy	SIECLOCK	
EIM_D25	ALT0	emi	WEIM_D[25]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	COL[6]	
	ALT2	usboh3	USBOTG_DATA1	
	ALT3	uart3	RXD_MUX	
	ALT4	uart2	CTS	
	ALT5	gpt	CMPOUT1	
	ALT6	tpiu	TRACE[9]	
	ALT7	usbphy	VBUSVALID	
EIM_D26	ALT0	emi	WEIM_D[26]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	COL[7]	
	ALT2	usboh3	USBOTG_DATA2	
	ALT3	uart3	TXD_MUX	
	ALT4	uart2	RTS	
	ALT5	gpt	CMPOUT2	
	ALT6	tpiu	TRACE[10]	
	ALT7	usbphy	AVALID	
EIM_D27	ALT0	emi	WEIM_D[27]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[9]	
	ALT2	usboh3	USBOTG_DATA3	
	ALT3	uart3	RTS	
	ALT4	i2c2	SCL	
	ALT5	audmux	AUD6_RXC	
	ALT6	tpiu	TRACE[11]	
	ALT7	usbphy	BVALID	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_D28	ALT0	emi	WEIM_D[28]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	ROW[4]	
	ALT2	usboh3	USBOTG_DATA4	
	ALT3	elvis_observe_mux	OBSRV_INT_OUT0	
	ALT4	ipu	SISG[0]	
	ALT5	audmux	AUD6_TXD	
	ALT6	tpiu	TRACE[12]	
	ALT7	usbphy	ENDSESSION	
EIM_D29	ALT0	emi	WEIM_D[29]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	ROW[5]	
	ALT2	usboh3	USBOTG_DATA5	
	ALT3	elvis_observe_mux	OBSRV_INT_OUT1	
	ALT4	ipu	SISG[1]	
	ALT5	audmux	AUD6_RXD	
	ALT6	tpiu	TRACE[13]	
	ALT7	usbphy	IDDIG	
EIM_D30	ALT0	emi	WEIM_D[30]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	ROW[6]	
	ALT2	usboh3	USBOTG_DATA6	
	ALT3	elvis_observe_mux	OBSRV_INT_OUT2	
	ALT4	ipu	SISG[2]	
	ALT5	audmux	AUD6_TXC	
	ALT6	tpiu	TRACE[14]	
	ALT7	usbphy	HOSTDISCONNECT	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_D31	ALT0	emi	WEIM_D[31]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	kpp	ROW[7]	
	ALT2	usboh3	USBOTG_DATA7	
	ALT3	elvis_observe_mux	OBSRV_INT_OUT3	
	ALT4	ipu	SISG[3]	
	ALT5	audmux	AUD6_TXFS	
	ALT6	tpiu	TRACE[15]	
EIM_A16	ALT0	emi	EIM_A[16]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[10]	
	ALT2	—	DATA_HS_OUT[0]	
	ALT7	src	OSC_FREQ_SEL[0]	
EIM_A17	ALT0	emi	EIM_A[17]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[11]	
EIM_A17	ALT2	—	DATA_HS_OUT[1]	
	ALT7	src	OSC_FREQ_SEL[1]	
EIM_A18	ALT0	emi	EIM_A[18]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[12]	
	ALT2	—	DATA_HS_OUT[2]	
	ALT7	src	BT_LPB[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_A19	ALT0	emi	EIM_A[19]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[13]	
	ALT2	—	DATA_HS_OUT[3]	
	ALT7	src	BT_LPB[1]	
EIM_A20	ALT0	emi	EIM_A[20]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[14]	
EIM_A20	ALT2	—	DATA_HS_OUT[4]	
	ALT7	src	BT_UART_SRC[0]	
EIM_A21	ALT0	emi	EIM_A[21]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[15]	
	ALT2	—	DATA_HS_OUT[5]	
	ALT7	src	BT_UART_SRC[1]	
EIM_A22	ALT0	emi	EIM_A[22]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[16]	
	ALT2	—	DATA_HS_OUT[6]	
EIM_A23	ALT0	emi	EIM_A[23]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[17]	
	ALT2	—	DATA_HS_OUT[7]	
EIM_A23	ALT7	src	BT_HPN_EN	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_A24	ALT0	emi	EIM_A[24]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[18]	
	ALT2	usboh3	USBH2_CLK	
EIM_A25	ALT0	emi	EIM_A[25]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[19]	
EIM_A25	ALT2	usboh3	USBH2_DIR	
	ALT6	ipu	DI1_PIN4	
EIM_A26	ALT0	emi	EIM_A[26]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[20]	
	ALT2	usboh3	USBH2_STP	
	ALT4	ipu	SISG[0]	
	ALT5	—	CSI1_DATA_EN	
	ALT6	ccm	DI2_EXT_CLK	
EIM_A27	ALT0	emi	EIM_A[27]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[21]	
	ALT2	usboh3	USBH2_NXT	
	ALT3	elvis_observe_mux	OBSRV_INT_OUT4	
	ALT4	ipu	SISG[1]	
	ALT5	—	CSI2_DATA_EN	
	ALT6	—	DI1_PIN1	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_EB0	ALT0	emi	EIM_EB[0]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT2	—	DETECT_D	
EIM_EB1	ALT0	emi	EIM_EB[1]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT2	—	DELAY_D	
EIM_EB2	ALT0	emi	EIM_EB[2]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[22]	
	ALT2	tpiu	TRCTL	
	ALT3	fec	MDIO	
	ALT4	ipu	SISG[2]	
	ALT5	—	CSI1_D[2]	
	ALT6	audmux	AUD5_RXFS	
	ALT7	gpt	CMPOUT1	
EIM_EB3	ALT0	emi	EIM_EB[3]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[23]	
	ALT2	tpiu	TRCLK	
	ALT3	fec	RDATA[1]	
	ALT4	ipu	SISG[3]	
	ALT5	—	CSI1_D[3]	
	ALT6	audmux	AUD5_RXC	
	ALT7	gpt	CMPOUT2	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_OE	ALT0	emi	EIM_OE	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[24]	
EIM_CS0	ALT0	emi	EIM_CS0	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[25]	
EIM_CS1	ALT0	emi	EIM_CS1	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[26]	
	ALT4	ipu	SISG[4]	
EIM_CS2	ALT0	emi	EIM_CS2	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[27]	
	ALT2	usboh3	USBOTG_STP	
EIM_CS2	ALT3	fec	RDATA[2]	
	ALT4	ipu	SISG[5]	
	ALT5	—	CSI1_D[4]	
	ALT6	audmux	AUD5_TXD	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_CS3	ALT0	emi	EIM_CS3	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[28]	
	ALT2	usboh3	USBOTG_NXT	
	ALT3	fec	RDATA[3]	
	ALT4	ccm	SSI_EXT2_CLK	
	ALT5	—	CSI1_D[5]	
	ALT6	audmux	AUD5_RXD	
EIM_CS4	ALT0	emi	EIM_CS4	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[29]	
	ALT2	usboh3	USBOTG_CLK	
	ALT3	fec	RX_ER	
	ALT4	ccm	SSI_EXT1_CLK	
	ALT5	—	CSI1_D[6]	
	ALT6	audmux	AUD5_TXC	
EIM_CS5	ALT0	emi	EIM_CS5	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio2	GPIO[30]	
	ALT2	usboh3	USBOTG_DIR	
	ALT3	fec	CRS	
	ALT4	ccm	DI1_EXT_CLK	
	ALT5	—	CSI1_D[7]	
	ALT6	audmux	AUD5_TXFS	
EIM_DTACK	ALT0	emi	WEIM_DTACK_B	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	gpio2	GPIO[31]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_WAIT	No Muxing (ALT0)	emi	EIM_WAIT	Drive Strength—Low Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
EIM_LBA	ALT0	emi	EIM_LBA	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio3	GPIO[1]	
EIM_BCLK	No Muxing (ALT0)	emi	EIM_BCLK	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
EIM_RW	No Muxing (ALT0)	emi	EIM_RW	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
EIM_CRE	ALT0	emi	EIM_CRE	—
	ALT1	gpio3	GPIO[2]	—

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_A0	No Muxing (ALT0)	emi	DRAM_A[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A1		emi	DRAM_A[1]	
DRAM_A2	No Muxing (ALT0)	emi	DRAM_A[2]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A3		emi	DRAM_A[3]	
DRAM_A4	No Muxing (ALT0)	emi	DRAM_A[4]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A5		emi	DRAM_A[5]	
DRAM_A6	No Muxing (ALT0)	emi	DRAM_A[6]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A7		emi	DRAM_A[7]	
DRAM_A8	No Muxing (ALT0)	emi	DRAM_A[8]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A9		emi	DRAM_A[9]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_A10	No Muxing (ALT0)	emi	DRAM_A[10]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A11		emi	DRAM_A[11]	
DRAM_A12	No Muxing (ALT0)	emi	DRAM_A[12]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_A13		emi	DRAM_A[13]	
DRAM_A14	No Muxing (ALT0)	emi	DRAM_A[14]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
EIM_SDBA1		emi	DRAM_SDBA[1]	
EIM_SDBA0	No Muxing (ALT0)	emi	DRAM_SDBA[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_RAS		emi	DRAM_RAS	
DRAM_CAS	No Muxing (ALT0)	emi	DRAM_CAS	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_SDWE		emi	DRAM_SDWE	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_SDCKE0	No Muxing (ALT0)	emi	DRAM_SDCKE[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_SDCKE1		emi	DRAM_SDCKE[1]	
DRAM_SDCLK	No Muxing (ALT0)	emi	DRAM_SDCLK	Drive Strength—CFG(High) Hyst. Enable—NA Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_SDCLK_B		emi	DRAM_SDCLK_B	Drive Strength—Low Hyst. Enable—NA Pull/Keep Select—NA Pull Up/Down Config.—NA dse test—regular Pull/Keep Enable—NA Slew Rate—NA test_ts—Disabled
DRAM_SDQS0		emi	DRAM_SDQS[0]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PD) dse test—regular Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_SDQS0_B	No Muxing (ALT0)	emi	DRAM_SDQS_B[0]	Drive Strength—NA Hyst. Enable—NA Pull/Keep Select—NA Pull Up/Down Config.—NA dse test—NA Pull/Keep Enable—NA Slew Rate—NA test_ts—Disabled
DRAM_SDQS1		emi	DRAM_SDQS[1]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PD) dse test—regular Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_SDQS1_B		emi	DRAM_SDQS_B[1]	Drive Strength—NA Hyst. Enable—NA Pull/Keep Select—NA Pull Up/Down Config.—NA dse test—NA Pull/Keep Enable—NA Slew Rate—NA test_ts—Disabled
DRAM_SDQS2	No Muxing (ALT0)	emi	DRAM_SDQS[2]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PD) dse test—regular Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_SDQS2_B		emi	DRAM_SDQS_B[2]	Drive Strength—NA Hyst. Enable—NA Pull/Keep Select—NA Pull Up/Down Config.—NA dse test—NA Pull/Keep Enable—NA Slew Rate—NA test_ts—Disabled
DRAM_SDQS3		emi	DRAM_SDQS[3]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PD) dse test—regular Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_SDQS3_B	No Muxing (ALT0)	emi	DRAM_SDQS_B[3]	Drive Strength—NA Hyst. Enable—NA Pull/Keep Select—NA Pull Up/ Down Config.—NA dse test—NA Pull/Keep Enable—NA Slew Rate—NA test_ts—Disabled
DRAM_CS0		emi	DRAM_CS0	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/ Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_CS1	ALT0	emi	DRAM_CS1	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/ Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—Controlled by the DRAM_CS0 SRE bit. test_ts—Disabled
	ALT1	ccm	CLKO	
DRAM_D0	No Muxing (ALT0)	emi	DRAM_D[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/ Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D1	No Muxing (ALT0)	emi	DRAM_D[1]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/ Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D2		emi	DRAM_D[2]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_D3	No Muxing (ALT0)	emi	DRAM_D[3]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D4		emi	DRAM_D[4]	
DRAM_D5	No Muxing (ALT0)	emi	DRAM_D[5]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D6		emi	DRAM_D[6]	
DRAM_D7	No Muxing (ALT0)	emi	DRAM_D[7]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D8		emi	DRAM_D[8]	
DRAM_D9	No Muxing (ALT0)	emi	DRAM_D[9]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D10		emi	DRAM_D[10]	
DRAM_D11	No Muxing (ALT0)	emi	DRAM_D[11]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D12		emi	DRAM_D[12]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_D13	No Muxing (ALT0)	emi	DRAM_D[13]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D14		emi	DRAM_D[14]	
DRAM_D15	No Muxing (ALT0)	emi	DRAM_D[15]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D16		emi	DRAM_D[16]	
DRAM_D17	No Muxing (ALT0)	emi	DRAM_D[17]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D18		emi	DRAM_D[18]	
DRAM_D19	No Muxing (ALT0)	emi	DRAM_D[19]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D20		emi	DRAM_D[20]	
DRAM_D21	No Muxing (ALT0)	emi	DRAM_D[21]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D22		emi	DRAM_D[22]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_D23	No Muxing (ALT0)	emi	DRAM_D[23]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D24		emi	DRAM_D[24]	
DRAM_D25	No Muxing (ALT0)	emi	DRAM_D[25]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D26		emi	DRAM_D[26]	
DRAM_D27	No Muxing (ALT0)	emi	DRAM_D[27]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D28		emi	DRAM_D[28]	
DRAM_D29	No Muxing (ALT0)	emi	DRAM_D[29]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_D30		emi	DRAM_D[30]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DRAM_D31	No Muxing (ALT0)	emi	DRAM_D[31]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—CFG(50Ohm) Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_DQM0		emi	DRAM_DQM[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_DQM1	No Muxing (ALT0)	emi	DRAM_DQM[1]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DRAM_DQM2		emi	DRAM_DQM[2]	
DRAM_DQM3	No Muxing (ALT0)	emi	DRAM_DQM[3]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
NANDF_WE_B	ALT0	emi	NANDF_WE_B	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DIOW	
	ALT2	esdhc3	DAT0	
	ALT3	gpio3	GPIO[3]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_RE_B	ALT0	emi	NANDF_RE_B	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DIOR	
	ALT2	esdhc3	DAT1	
	ALT3	gpio3	GPIO[4]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[1]	
NANDF_ALE	ALT0	emi	NANDF_ALE	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	BUFFER_EN	
	ALT3	gpio3	GPIO[5]	
NANDF_ALE	ALT4	sdma	DEBUG_EVT_CHN_LINES[2]	
NANDF_CLE	ALT0	emi	NANDF_CLE	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_RESET_B	
	ALT3	gpio3	GPIO[6]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[3]	
NANDF_WP_B	ALT0	emi	NANDF_WP_B	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DMACK	
	ALT2	esdhc3	DAT2	
NANDF_WP_B	ALT3	gpio3	GPIO[7]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[4]	
NANDF_RB0	ALT0	emi	NANDF_RB0	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DMARQ	
	ALT2	esdhc3	DAT3	
	ALT3	gpio3	GPIO[8]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[4]	
	ALT5	ecspi2	SS1	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_RB1	ALT0	emi	NANDF_RB1	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	IORDY	
	ALT2	ecspi2	RDY	
	ALT3	gpio3	GPIO[9]	
	ALT4	gpt	CMPOUT2	
	ALT5	esdhc4	CMD	
	ALT6	cspi	MOSI	
NANDF_RB2	ALT0	emi	NANDF_RB2	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
NANDF_RB2	ALT1	fec	COL	
	ALT2	ecspi2	SCLK	
	ALT3	gpio3	GPIO[10]	
	ALT4	gpt	CMPOUT3	
	ALT5	—	DI2_WAIT	
	ALT6	usboh3	USBH3_NXT	
	ALT7	usboh3	H3_DP	
NANDF_RB3	ALT0	emi	NANDF_RB3	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	fec	RX_CLK	
	ALT2	ecspi2	MISO	
	ALT3	gpio3	GPIO[11]	
	ALT4	dpllip1	TOG_EN	
	ALT5	—	DI1_WAIT	
	ALT6	usboh3	USBH3_CLK	
	ALT7	usboh3	H3_DM	
EIM_SDBA2	No Muxing (ALT0)	emi	DRAM_SDBA[2]	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
EIM_SDODT1		emi	DRAM_ODT1	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
EIM_SDODT0	No Muxing (ALT0)	emi	DRAM_ODT0	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) odt—NA Pull Up/Down Config.—CFG(100 KΩ PU) DDR / CMOS Input Mode—CFG(CMOS) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
GPIO_NAND	ALT0	gpio3	GPIO[12]	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	INTRQ	
NANDF_CS0	ALT0	emi	NANDF_CS0	Drive Strength—CFG(High) low/high output voltage—NA Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT3	gpio3	GPIO[16]	
NANDF_CS1	ALT0	emi	NANDF_CS1	Drive Strength—CFG(High) low/high output voltage—NA Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT3	gpio3	GPIO[17]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[5]	
NANDF_CS2	ALT0	emi	NANDF_CS2	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	CS_0	
	ALT2	fec	TX_ER	
	ALT3	gpio3	GPIO[18]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[5]	
	ALT5	esdhc4	CLK	
	ALT6	cspi	SCLK	
	ALT7	usboh3	H1_DP	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_CS3	ALT0	emi	NANDF_CS3	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	CS_1	
	ALT2	fec	MDC	
	ALT3	gpio3	GPIO[19]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[6]	
	ALT5	esdhc4	DAT0	
	ALT6	sim	PD0	
	ALT7	usboh3	H1_DM	
NANDF_CS4	ALT0	emi	NANDF_CS4	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DA_0	
	ALT2	fec	TDATA[1]	
	ALT3	gpio3	GPIO[20]	
	ALT4	sdma	DEBUG_EVT_CHN_LINES[7]	
	ALT5	esdhc4	DAT1	
	ALT6	sim	CLK0	
	ALT7	usboh3	USBH3_STP	
NANDF_CS5	ALT0	emi	NANDF_CS5	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DA_1	
	ALT2	fec	TDATA[2]	
	ALT3	gpio3	GPIO[21]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[0]	
	ALT5	esdhc4	DAT2	
	ALT6	sim	RST0	
	ALT7	usboh3	USBH3_DIR	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_CS6	ALT0	emi	NANDF_CS6	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(360 KΩ PD) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	DA_2	
	ALT2	fec	TDATA[3]	
	ALT3	gpio3	GPIO[22]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[1]	
	ALT5	esdhc4	DAT3	
	ALT6	sim	VEN0	
	ALT7	cspi	SS3	
NANDF_CS7	ALT0	emi	NANDF_CS7	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	fec	TX_EN	
	ALT3	gpio3	GPIO[23]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[2]	
	ALT5	esdhc3	CLK	
	ALT6	sim	TX0	
NANDF_RDY_INT	ALT0	emi	RDY	Drive Strength—CFG(Low) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	fec	TX_CLK	
NANDF_RDY_INT	ALT2	ecspi2	SS0	
	ALT3	gpio3	GPIO[24]	
	ALT4	sdma	DEBUG_EVENT_CHANNEL[3]	
	ALT5	esdhc3	CMD	
	ALT6	sim	RX0	
NANDF_D15	ALT0	emi	EIM_NFC_D[15]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[15]	
	ALT2	ecspi2	MOSI	
	ALT3	gpio3	GPIO[25]	
	ALT4	sdma	DEBUG_PC[0]	
	ALT5	esdhc3	DAT7	
	ALT6	ipu	IPU_DIAG_BUS[0]	
	ALT7	gpu3d	GPU_DEBUG_OUT[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_D14	ALT0	emi	EIM_NFC_D[14]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[14]	
	ALT2	ecspi2	SS3	
	ALT3	gpio3	GPIO[26]	
	ALT4	sdma	DEBUG_PC[1]	
	ALT5	esdhc3	DAT6	
	ALT6	ipu	IPU_DIAG_BUS[1]	
	ALT7	gpu3d	GPU_DEBUG_OUT[1]	
NANDF_D13	ALT0	emi	EIM_NFC_D[13]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[13]	
	ALT2	ecspi2	SS2	
	ALT3	gpio3	GPIO[27]	
	ALT4	sdma	DEBUG_PC[2]	
	ALT5	esdhc3	DAT5	
	ALT6	ipu	IPU_DIAG_BUS[2]	
	ALT7	gpu3d	GPU_DEBUG_OUT[2]	
NANDF_D12	ALT0	emi	EIM_NFC_D[12]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[12]	
	ALT2	ecspi2	SS1	
	ALT3	gpio3	GPIO[28]	
	ALT4	sdma	DEBUG_PC[3]	
	ALT5	esdhc3	DAT4	
	ALT6	ipu	IPU_DIAG_BUS[3]	
	ALT7	gpu3d	GPU_DEBUG_OUT[3]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_D11	ALT0	emi	EIM_NFC_D[11]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[11]	
	ALT2	fec	RX_DV	
	ALT3	gpio3	GPIO[29]	
	ALT4	sdma	DEBUG_PC[4]	
	ALT5	esdhc3	DAT3	
	ALT6	ipu	IPU_DIAG_BUS[4]	
	ALT7	gpu3d	GPU_DEBUG_OUT[4]	
NANDF_D10	ALT0	emi	EIM_NFC_D[10]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[10]	
	ALT3	gpio3	GPIO[30]	
	ALT4	sdma	DEBUG_PC[5]	
	ALT5	esdhc3	DAT2	
	ALT6	ipu	IPU_DIAG_BUS[5]	
	ALT7	gpu3d	GPU_DEBUG_OUT[5]	
NANDF_D9	ALT0	emi	EIM_NFC_D[9]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
NANDF_D9	ALT1	pata	PATA_DATA[9]	
	ALT2	fec	RDATA[0]	
	ALT3	gpio3	GPIO[31]	
	ALT4	sdma	DEBUG_PC[6]	
	ALT5	esdhc3	DAT1	
	ALT6	ipu	IPU_DIAG_BUS[6]	
	ALT7	gpu3d	GPU_DEBUG_OUT[6]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_D8	ALT0	emi	EIM_NFC_D[8]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[8]	
	ALT2	fec	TDATA[0]	
	ALT3	gpio4	GPIO[0]	
	ALT4	sdma	DEBUG_PC[7]	
	ALT5	esdhc3	DAT0	
	ALT6	ipu	IPU_DIAG_BUS[7]	
	ALT7	gpu3d	GPU_DEBUG_OUT[7]	
NANDF_D7	ALT0	emi	EIM_NFC_D[7]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[7]	
	ALT3	gpio4	GPIO[1]	
	ALT4	sdma	DEBUG_PC[8]	
	ALT5	usboh3	USBH3_DATA0	
	ALT6	ipu	IPU_DIAG_BUS[8]	
	ALT7	gpu3d	GPU_DEBUG_OUT[8]	
NANDF_D6	ALT0	emi	EIM_NFC_D[6]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
NANDF_D6	ALT1	pata	PATA_DATA[6]	
	ALT2	esdhc4	LCTL	
	ALT3	gpio4	GPIO[2]	
	ALT4	sdma	DEBUG_PC[9]	
	ALT5	usboh3	USBH3_DATA1	
	ALT6	ipu	IPU_DIAG_BUS[9]	
	ALT7	gpu3d	GPU_DEBUG_OUT[9]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_D5	ALT0	emi	EIM_NFC_D[5]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[5]	
	ALT2	esdhc4	WP	
	ALT3	gpio4	GPIO[3]	
	ALT4	sdma	DEBUG_PC[10]	
	ALT5	usboh3	USBH3_DATA2	
	ALT6	ipu	IPU_DIAG_BUS[10]	
	ALT7	gpu3d	GPU_DEBUG_OUT[10]	
NANDF_D4	ALT0	emi	EIM_NFC_D[4]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[4]	
	ALT2	esdhc4	CD	
	ALT3	gpio4	GPIO[4]	
	ALT4	sdma	DEBUG_PC[11]	
	ALT5	usboh3	USBH3_DATA3	
	ALT6	ipu	IPU_DIAG_BUS[11]	
	ALT7	gpu3d	GPU_DEBUG_OUT[11]	
NANDF_D3	ALT0	emi	EIM_NFC_D[3]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[3]	
	ALT2	esdhc4	DAT4	
	ALT3	gpio4	GPIO[5]	
	ALT4	sdma	DEBUG_PC[12]	
	ALT5	usboh3	USBH3_DATA4	
	ALT6	ipu	IPU_DIAG_BUS[12]	
	ALT7	gpu3d	GPU_DEBUG_OUT[12]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
NANDF_D2	ALT0	emi	EIM_NFC_D[2]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[2]	
	ALT2	esdhc4	DAT5	
	ALT3	gpio4	GPIO[6]	
	ALT4	sdma	DEBUG_PC[13]	
	ALT5	usboh3	USBH3_DATA5	
	ALT6	ipu	IPU_DIAG_BUS[13]	
	ALT7	gpu3d	GPU_DEBUG_OUT[13]	
NANDF_D1	ALT0	emi	EIM_NFC_D[1]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[1]	
	ALT2	esdhc4	DAT6	
	ALT3	gpio4	GPIO[7]	
	ALT4	sdma	DEBUG_CORE_STATE[0]	
	ALT5	usboh3	USBH3_DATA6	
	ALT6	ipu	IPU_DIAG_BUS[14]	
	ALT7	gpu3d	GPU_DEBUG_OUT[14]	
NANDF_D0	ALT0	emi	EIM_NFC_D[0]	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	pata	PATA_DATA[0]	
	ALT2	esdhc4	DAT7	
	ALT3	gpio4	GPIO[8]	
	ALT4	sdma	DEBUG_CORE_STATE[1]	
	ALT5	usboh3	USBH3_DATA7	
	ALT6	ipu	IPU_DIAG_BUS[15]	
	ALT7	gpu3d	GPU_DEBUG_OUT[15]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSI1_D8	ALT0	—	CSI1_D[8]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DETECT_Z	
	ALT3	gpio3	GPIO[12]	
CSI1_D9	ALT0	—	CSI1_D[9]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DELAY_Z	
	ALT3	gpio3	GPIO[13]	
CSI1_D10	No Muxing (ALT0)	—	CSI1_D[10]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI1_D11		—	CSI1_D[11]	
CSI1_D12	No Muxing (ALT0)	—	CSI1_D[12]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI1_D13		—	CSI1_D[13]	
CSI1_D14	No Muxing (ALT0)	—	CSI1_D[14]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI1_D15		—	CSI1_D[15]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSI1_D16	No Muxing (ALT0)	—	CSI1_D[16]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI1_D17		—	CSI1_D[17]	
CSI1_D18	No Muxing (ALT0)	—	CSI1_D[18]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI1_D19		—	CSI1_D[19]	
CSI1_VSYNC	ALT0	—	CSI1_VSYNC	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	TX_DDR_Q	
	ALT3	gpio3	GPIO[14]	
CSI1_HSYNC	ALT0	—	CSI1_HSYNC	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	TX_DDR_I	
	ALT3	gpio3	GPIO[15]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSI1_PIXCLK	No Muxing (ALT0)	—	CSI1_PIXCLK	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—FAST test_ts—Disabled
CSI1_MCLK		ccm	CSI1_MCLK	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
CSI2_D12	ALT0	—	CSI2_D[12]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	LP_RX_E	
	ALT3	gpio4	GPIO[9]	
CSI2_D13	ALT0	—	CSI2_D[13]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	RX_VALID_ESC_OUT	
	ALT3	gpio4	GPIO[10]	
CSI2_D13	ALT6	emi	EMI_DEBUG[45]	
CSI2_D14	No Muxing (ALT0)	—	CSI2_D[14]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI2_D15		—	CSI2_D[15]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSI2_D16	No Muxing (ALT0)	—	CSI2_D[16]	Drive Strength—Low Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
CSI2_D17		—	CSI2_D[17]	
CSI2_D18	ALT0	—	CSI2_D[18]	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	—	HS_TX_E	
	ALT2	elvis_observe_mux	OBSRV_INT_OUT0	
	ALT3	gpio4	GPIO[11]	
	ALT4	sdma	DEBUG_RTBUFFER_WRITE	
	ALT6	emi	EMI_DEBUG[46]	
CSI2_D19	ALT0	—	CSI2_D[19]	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	—	LP_TX_E	
	ALT2	elvis_observe_mux	OBSRV_INT_OUT1	
	ALT3	gpio4	GPIO[12]	
	ALT4	sdma	DEBUG_YIELD	
	ALT6	emi	EMI_DEBUG[47]	
CSI2_VSYNC	ALT0	—	CSI2_VSYNC	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	—	HS_RX_E	
	ALT2	elvis_observe_mux	OBSRV_INT_OUT2	
	ALT3	gpio4	GPIO[13]	
	ALT4	sdma	DEBUG_BUS_ERROR	
	ALT6	emi	EMI_DEBUG[48]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSI2_HSYNC	ALT0	—	CSI2_HSYNC	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	TX_BYTE_CLK_HS_OUT	
	ALT2	elvis_observe_mux	OBSRV_INT_OUT3	
	ALT3	gpio4	GPIO[14]	
	ALT6	emi	EMI_DEBUG[49]	
CSI2_PIXCLK	ALT0	—	CSI2_PIXCLK	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
CSI2_PIXCLK	ALT1	—	RX_BYTE_CLK_HS_OUT	
	ALT2	elvis_observe_mux	OBSRV_INT_OUT4	
	ALT3	gpio4	GPIO[15]	
	ALT6	emi	EMI_DEBUG[50]	
I2C1_CLK	ALT0	hsi2c	SCL	Drive Strength—CFG(High) low/high output voltage—NA Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT3	gpio4	GPIO[16]	
I2C1_DAT	ALT0	hsi2c	SDA	Drive Strength—CFG(High) low/high output voltage—NA Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT3	gpio4	GPIO[17]	
AUD3_BB_TXD	ALT0	audmux	AUD3_TXD	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	slm	DATA	
	ALT3	gpio4	GPIO[18]	
	ALT7	usbphy	DATAOUT[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
AUD3_BB_RXD	ALT0	audmux	AUD3_RXD	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart3	RXD_MUX	
AUD3_BB_RXD	ALT3	gpio4	GPIO[19]	
	ALT7	usbphy	DATAOUT[1]	
AUD3_BB_CK	ALT0	audmux	AUD3_TXC	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	slm	CLK	
	ALT3	gpio4	GPIO[20]	
	ALT7	usbphy	DATAOUT[2]	
AUD3_BB_FS	ALT0	audmux	AUD3_TXFS	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart3	TXD_MUX	
	ALT3	gpio4	GPIO[21]	
	ALT7	usbphy	DATAOUT[3]	
CSPI1_MOSI	ALT0	ecspi1	MOSI	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	i2c1	SDA	
CSPI1_MOSI	ALT3	gpio4	GPIO[22]	
	ALT7	usbphy	DATAOUT[4]	
CSPI1_MISO	ALT0	ecspi1	MISO	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	audmux	AUD4_RXD	
	ALT3	gpio4	GPIO[23]	
	ALT7	usbphy	DATAOUT[5]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CSPI1_SS0	ALT0	ecspi1	SS0	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	audmux	AUD4_TXC	
	ALT3	gpio4	GPIO[24]	
	ALT7	usbphy	DATAOUT[6]	
CSPI1_SS1	ALT0	ecspi1	SS1	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	audmux	AUD4_TXD	
CSPI1_SS1	ALT3	gpio4	GPIO[25]	
	ALT7	usbphy	DATAOUT[7]	
CSPI1_RDY	ALT0	ecspi1	RDY	Drive Strength—CFG(Low) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	audmux	AUD4_TXFS	
	ALT3	gpio4	GPIO[26]	
	ALT7	usbphy	DATAOUT[8]	
CSPI1_SCLK	ALT0	ecspi1	SCLK	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	i2c1	SCL	
	ALT3	gpio4	GPIO[27]	
	ALT7	usbphy	DATAOUT[9]	
UART1_RXD	ALT0	uart1	RXD_MUX	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT3	gpio4	GPIO[28]	
UART1_RXD	ALT5	—	READY_ESC_OUT	
	ALT6	emi	EMI_DEBUG[12]	
	ALT7	usbphy	DATAOUT[10]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
UART1_TXD	ALT0	uart1	TXD_MUX	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	pwm2	PWMO	
	ALT3	gpio4	GPIO[29]	
	ALT5	—	REQUEST_ESC_OUT	
	ALT6	emi	EMI_DEBUG[13]	
	ALT7	usbphy	DATAOUT[11]	
UART1_RTS	ALT0	uart1	RTS	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT3	gpio4	GPIO[30]	
	ALT5	—	CLK_ESC_OUT	
	ALT6	emi	EMI_DEBUG[14]	
	ALT7	usbphy	DATAOUT[12]	
UART1_CTS	ALT0	uart1	CTS	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
UART1_CTS	ALT3	gpio4	GPIO[31]	
	ALT5	—	READY_HS_OUT	
	ALT6	emi	EMI_DEBUG[15]	
	ALT7	usbphy	DATAOUT[13]	
UART2_RXD	ALT0	uart2	RXD_MUX	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	firi	TXD	
	ALT3	gpio1	GPIO[20]	
	ALT5	—	VALID_HS_OUT	
	ALT6	emi	EMI_DEBUG[16]	
	ALT7	usbphy	DATAOUT[14]	
UART2_TXD	ALT0	uart2	TXD_MUX	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	firi	RXD	
	ALT3	gpio1	GPIO[21]	
	ALT5	—	VALID_ESC_OUT	
	ALT6	emi	EMI_DEBUG[17]	
	ALT7	usbphy	DATAOUT[15]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
UART3_RXD	ALT0	uart1	DTR	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart3	RXD_MUX	
	ALT2	—	CSI1_D[0]	
	ALT3	gpio1	GPIO[22]	
	ALT5	—	SYNC_HS_OUT	
	ALT6	emi	EMI_DEBUG[18]	
	ALT7	usbphy	LINESTATE[0]	
UART3_TXD	ALT0	uart1	DSR	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart3	TXD_MUX	
	ALT2	—	CSI1_D[1]	
	ALT3	gpio1	GPIO[23]	
	ALT5	—	REQUEST_HS_OUT	
	ALT6	emi	EMI_DEBUG[19]	
	ALT7	usbphy	LINESTATE[1]	
OWIRE_LINE	ALT0	owire	LINE	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—Pull Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT3	gpio1	GPIO[24]	
	ALT5	tigerp_platform_ne_32k_256k	CTI_TRIGOUT6	
	ALT6	spdif	OUT1	
	ALT7	src	SYSTEM_RST	
KEY_ROW0	ALT0	kpp	ROW[0]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
KEY_ROW0	ALT1	emi	DSTROBE	
	ALT2	—	TX_DDR_Q	
	ALT5	scc	RANDOM_V	
	ALT6	emi	EMI_DEBUG[20]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
KEY_ROW1	ALT0	kpp	ROW[1]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	rtic	RTIC_SEC_VIO	
	ALT2	—	TX_DDR_I	
	ALT5	scc	RANDOM	
	ALT6	emi	EMI_DEBUG[21]	
KEY_ROW2	ALT0	kpp	ROW[2]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	rtic	RTIC_DONE_INT	
	ALT2	—	TX_DDR_Q	
	ALT5	sjc	DONE	
	ALT6	emi	EMI_DEBUG[22]	
KEY_ROW3	ALT0	kpp	ROW[3]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
KEY_ROW3	ALT1	csu	CSU_ALARM_AUT[0]	
	ALT2	—	TX_DDR_I	
	ALT5	sjc	FAIL	
	ALT6	emi	EMI_DEBUG[23]	
KEY_COL0	ALT0	kpp	COL[0]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	csu	CSU_ALARM_AUT[1]	
	ALT2	—	HS_TX_E0	
	ALT7	ccm	PLL1_BYP	
KEY_COL1	ALT0	kpp	COL[1]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	csu	CSU_ALARM_AUT[2]	
	ALT2	—	HS_TX_E1	
	ALT7	ccm	PLL2_BYP	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
KEY_COL2	ALT0	kpp	COL[2]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	ipu	SNOOP2	
KEY_COL2	ALT7	ccm	PLL3_BYP	
KEY_COL3	ALT0	kpp	COL[3]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT5	tigerp_platform_ne_32k_256k	CTI_TRIGOUT7	
	ALT7	src	INT_BOOT	
KEY_COL4	ALT0	kpp	COL[4]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart1	RI	
KEY_COL4	ALT2	uart3	RTS	
	ALT3	i2c2	SCL	
	ALT4	ccm	SSI_EXT2_CLK	
	ALT5	tigerp_platform_ne_32k_256k	CTI_TRIGIN_ACK7	
	ALT6	spdif	OUT1	
	ALT7	src	ANY_PU_RST	
KEY_COL5	ALT0	kpp	COL[5]	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart1	DCD	
	ALT2	uart3	CTS	
	ALT3	i2c2	SDA	
	ALT4	ccm	SSI_EXT1_CLK	
	ALT5	tigerp_platform_ne_32k_256k	CTI_TRIGIN7	
	ALT6	—	MCT_EXT_ACT_TRIG	
	ALT7	sjc	JTAG_ACT	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
JTAG_TCK	No Muxing (ALT0)	sjc	TCK	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PD Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(SLOW) test_ts—Disabled
JTAG_TMS		sjc	TMS	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—Pull Pull Up/Down Config.—47 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(SLOW) test_ts—Disabled
JTAG_TDI	No Muxing (ALT0)	sjc	TDI	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—Pull Pull Up/Down Config.—47 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(SLOW) test_ts—Disabled
JTAG_TDO		sjc	TDO	Drive Strength—High Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—FAST test_ts—Disabled

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
JTAG_TRSTB	No Muxing (ALT0)	sjc	TRSTB	Drive Strength—Low Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—47 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(SLOW) test_ts—Disabled
JTAG_DE_B		sjc	DE_B	Drive Strength—High Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—47 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Enabled Pull/Keep Enable—Enabled Slew Rate—FAST test_ts—Disabled
JTAG_MOD	No Muxing (ALT0)	sjc	MOD	Drive Strength—Low Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—CFG(SLOW) test_ts—Disabled
USBH1_CLK	ALT0	usboh3	USBH1_CLK	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	SCLK	
	ALT2	gpio1	GPIO[25]	
	ALT3	—	LPDT_ESC_OUT	
	ALT4	sdma	DEBUG_CORE_RUN	
USBH1_CLK	ALT5	i2c2	SCL	
	ALT6	emi	EMI_DEBUG[0]	
	ALT7	uart3	CTS	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
USBH1_DIR	ALT0	usboh3	USBH1_DIR	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	MOSI	
	ALT2	gpio1	GPIO[26]	
	ALT3	—	HS_RX_Z	
	ALT4	sdma	DEBUG_MODE	
	ALT5	i2c2	SDA	
	ALT6	emi	EMI_DEBUG[1]	
	ALT7	uart3	RTS	
USBH1_STP	ALT0	usboh3	USBH1_STP	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	RDY	
	ALT2	gpio1	GPIO[27]	
	ALT3	—	ACTIVE_HS_OUT	
	ALT4	sdma	DEBUG_EVENT_CHANNEL_SEL	
	ALT5	uart3	RXD_MUX	
	ALT6	emi	EMI_DEBUG[2]	
	ALT7	usbphy	BISTOK	
USBH1_NXT	ALT0	usboh3	USBH1_NXT	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	MISO	
	ALT2	gpio1	GPIO[28]	
	ALT3	—	RX_TERM_E	
	ALT4	sdma	DEBUG_BUS_RWB	
	ALT5	uart3	TXD_MUX	
	ALT6	emi	EMI_DEBUG[3]	
	ALT7	usbphy	ONBIST	
USBH1_DATA0	ALT0	usboh3	USBH1_DATA0	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart2	CTS	
	ALT2	gpio1	GPIO[11]	
	ALT3	—	DATA_ESC_OUT[0]	
	ALT4	sdma	DEBUG_CORE_STATE[2]	
	ALT6	emi	EMI_DEBUG[4]	
	ALT7	usbphy	VSTATUS[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
USBH1_DATA1	ALT0	usboh3	USBH1_DATA1	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart2	RXD_MUX	
	ALT2	gpio1	GPIO[12]	
	ALT3	—	DATA_ESC_OUT[1]	
	ALT4	sdma	DEBUG_CORE_STATE[3]	
	ALT6	emi	EMI_DEBUG[5]	
	ALT7	usbphy	VSTATUS[1]	
USBH1_DATA2	ALT0	usboh3	USBH1_DATA2	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart2	TXD_MUX	
	ALT2	gpio1	GPIO[13]	
	ALT3	—	DATA_ESC_OUT[2]	
	ALT4	sdma	DEBUG_MATCHED_DMBUS	
	ALT6	emi	EMI_DEBUG[6]	
	ALT7	usbphy	VSTATUS[2]	
USBH1_DATA3	ALT0	usboh3	USBH1_DATA3	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	uart2	RTS	
	ALT2	gpio1	GPIO[14]	
	ALT3	—	DATA_ESC_OUT[3]	
	ALT4	sdma	DEBUG_BUS_DEVICE[0]	
	ALT6	emi	EMI_DEBUG[7]	
	ALT7	usbphy	VSTATUS[3]	
USBH1_DATA4	ALT0	usboh3	USBH1_DATA4	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	SS0	
	ALT2	gpio1	GPIO[15]	
	ALT3	—	DATA_ESC_OUT[4]	
	ALT4	sdma	DEBUG_BUS_DEVICE[1]	
	ALT6	emi	EMI_DEBUG[8]	
	ALT7	usbphy	VSTATUS[4]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
USBH1_DATA5	ALT0	usboh3	USBH1_DATA5	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	SS1	
	ALT2	gpio1	GPIO[16]	
	ALT3	—	DATA_ESC_OUT[5]	
	ALT4	sdma	DEBUG_BUS_DEVICE[2]	
	ALT6	emi	EMI_DEBUG[9]	
	ALT7	usbphy	VSTATUS[5]	
USBH1_DATA6	ALT0	usboh3	USBH1_DATA6	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	cspi	SS3	
	ALT2	gpio1	GPIO[17]	
	ALT3	—	DATA_ESC_OUT[6]	
	ALT4	sdma	DEBUG_BUS_DEVICE[3]	
	ALT6	emi	EMI_DEBUG[10]	
	ALT7	usbphy	VSTATUS[6]	
USBH1_DATA7	ALT0	usboh3	USBH1_DATA7	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	ecspi1	SS3	
	ALT2	gpio1	GPIO[18]	
	ALT3	—	DATA_ESC_OUT[7]	
	ALT4	sdma	DEBUG_BUS_DEVICE[4]	
	ALT5	ecspi2	SS3	
	ALT6	emi	EMI_DEBUG[11]	
	ALT7	usbphy	VSTATUS[7]	
DI1_PIN11	ALT0	ipu	DI1_PIN11	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	READY_ESC_IN	
	ALT4	gpio3	GPIO[0]	
	ALT7	ecspi1	SS2	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DI1_PIN12	ALT0	ipu	DI1_PIN12	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	REQUEST_ESC_IN	
DI1_PIN12	ALT4	gpio3	GPIO[1]	
DI1_PIN13	ALT0	ipu	DI1_PIN13	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	CLK_ESC_IN	
	ALT4	gpio3	GPIO[2]	
DI1_D0_CS	ALT0	ipu	DI1_D0_CS	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	LPDT_ESC_IN	
DI1_D0_CS	ALT4	gpio3	GPIO[3]	
DI1_D1_CS	ALT0	ipu	DI1_D1_CS	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	READY_HS_IN	
	ALT2	ipu	DI1_PIN14	
	ALT3	ipu	DI1_PIN5	
	ALT4	gpio3	GPIO[4]	
DISPB2_SER_DIN	ALT0	ipu	DISPB2_SER_DIN	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	VALID_HS_IN	
	ALT2	—	DI1_PIN1	
	ALT4	gpio3	GPIO[5]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISPB2_SER_DIO	ALT0	ipu	DISPB2_SER_DIO	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	ACTIVE_HS_IN	
DISPB2_SER_DIO	ALT3	ipu	DI1_PIN6	
	ALT4	gpio3	GPIO[6]	
	ALT6	wdog1	WDOG_RST_B_DEB	
DISPB2_SER_CLK	ALT0	ipu	DISPB2_SER_CLK	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	BYTE_CLK_HS_IN	
	ALT2	ipu	DI1_PIN17	
	ALT3	ipu	DI1_PIN7	
	ALT4	gpio3	GPIO[7]	
	ALT6	wdog2	WDOG_RST_B_DEB	
DISPB2_SER_RS	ALT0	ipu	DISPB2_SER_RS	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	VALID_ESC_IN	
	ALT2	ipu	DI1_PIN16	
	ALT3	ipu	DI1_PIN8	
	ALT4	gpio3	GPIO[8]	
DISP1_DAT0	No Muxing (ALT0)	ipu	DISP1_DAT[0]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP1_DAT1	No Muxing (ALT0)	ipu	DISP1_DAT[1]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP1_DAT2		ipu	DISP1_DAT[2]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP1_DAT3	No Muxing (ALT0)	ipu	DISP1_DAT[3]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP1_DAT4		ipu	DISP1_DAT[4]	
DISP1_DAT5	No Muxing (ALT0)	ipu	DISP1_DAT[5]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP1_DAT6	ALT0	ipu	DISP1_DAT[6]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_USB_SRC	
DISP1_DAT7	ALT0	ipu	DISP1_DAT[7]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_EEPROM_CFG	
DISP1_DAT8	ALT0	ipu	DISP1_DAT[8]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_SRC[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP1_DAT9	ALT0	ipu	DISP1_DAT[9]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_SRC[1]	
DISP1_DAT10	ALT0	ipu	DISP1_DAT[10]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_SPARE_SIZE	
DISP1_DAT11	ALT0	ipu	DISP1_DAT[11]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_LPB_FREQ[2]	
DISP1_DAT12	ALT0	ipu	DISP1_DAT[12]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_MLC_SEL	
DISP1_DAT13	ALT0	ipu	DISP1_DAT[13]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_MEM_CTL[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP1_DAT14	ALT0	ipu	DISP1_DAT[14]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_MEM_CTL[1]	
DISP1_DAT15	ALT0	ipu	DISP1_DAT[15]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_BUS_WIDTH	
DISP1_DAT16	ALT0	ipu	DISP1_DAT[16]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_PAGE_SIZE[0]	
DISP1_DAT17	ALT0	ipu	DISP1_DAT[17]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT7	src	BT_PAGE_SIZE[1]	
DISP1_DAT18	ALT0	ipu	DISP1_DAT[18]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT4	ipu	DI2_PIN5	
	ALT5	ipu	DI2_PIN11	
	ALT7	src	BT_WEIM_MUXED[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP1_DAT19	ALT0	ipu	DISP1_DAT[19]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT4	ipu	DI2_PIN6	
	ALT5	ipu	DI2_PIN12	
	ALT7	src	BT_WEIM_MUXED[1]	
DISP1_DAT20	ALT0	ipu	DISP1_DAT[20]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT4	ipu	DI2_PIN7	
DISP1_DAT20	ALT5	ipu	DI2_PIN13	
	ALT7	src	BT_MEM_TYPE[0]	
DISP1_DAT21	ALT0	ipu	DISP1_DAT[21]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT4	ipu	DI2_PIN8	
	ALT5	ipu	DI2_PIN14	
	ALT7	src	BT_MEM_TYPE[1]	
DISP1_DAT22	ALT0	ipu	DISP1_DAT[22]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT3	tigerp_platform_ne_32k_256k	CTI_TRIGOUT_ACK6	
	ALT5	ipu	DISP2_DAT[16]	
	ALT6	ipu	DI2_D0_CS	
	ALT7	src	BT_LPB_FREQ[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP1_DAT23	ALT0	ipu	DISP1_DAT[23]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP1_DAT23	ALT3	tigerp_platform_ne_32k_256k	CTI_TRIGOUT_ACK7	
	ALT4	ipu	SER_DISP2_CS	
	ALT5	ipu	DISP2_DAT[17]	
	ALT6	ipu	DI2_D1_CS	
	ALT7	src	BT_LPB_FREQ[1]	
DI1_PIN3	ALT0	ipu	DI1_PIN3	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	TX_DDR_Q	
	ALT6	emi	EMI_DEBUG[24]	
	ALT7	sim	SIM_TX_CLK_TEST	
DI1_DISP_CLK	No Muxing (ALT0)	ipu	DI1_DISP_CLK	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DI1_PIN2	ALT0	ipu	DI1_PIN2	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	TX_DDR_I	
	ALT6	emi	EMI_DEBUG[25]	
	ALT7	sim	SIM_RCV_CLK_TEST	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DI1_PIN15	No Muxing (ALT0)	ipu	DI1_PIN15	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—Keep Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DI_GP1	ALT0	ipu	DISPB1_SER_RS	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	RX_VALID_ESC_IN	
	ALT2	ccm	DI1_EXT_CLK	
	ALT6	emi	EMI_DEBUG[26]	
DI_GP2	ALT0	ipu	DISPB1_SER_CLK	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	SYNC_HS_IN	
	ALT2	—	DI2_WAIT	
	ALT6	emi	EMI_DEBUG[27]	
DI_GP3	ALT0	ipu	DISPB1_SER_DIO	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	REQUEST_HS_IN	
DI_GP3	ALT2	fec	TX_ER	
	ALT3	—	CSI1_DATA_EN	
	ALT6	emi	EMI_DEBUG[28]	
DI2_PIN4	ALT0	ipu	DI2_PIN4	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[0]	
	ALT2	fec	CRS	
	ALT3	—	CSI2_DATA_EN	
	ALT6	emi	EMI_DEBUG[29]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DI2_PIN2	ALT0	ipu	DI2_PIN2	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[1]	
	ALT2	fec	MDC	
	ALT6	emi	EMI_DEBUG[30]	
DI2_PIN3	ALT0	ipu	DI2_PIN3	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—22 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[2]	
DI2_PIN3	ALT2	fec	MDIO	
	ALT6	emi	EMI_DEBUG[31]	
DI2_DISP_CLK	ALT0	ipu	DI2_DISP_CLK	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT2	fec	RDATA[1]	
DI_GP4	ALT0	ipu	DISPB1_SER_DIN	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(SLOW) test_ts—Disabled
	ALT1	—	DATA_HS_IN[3]	
DI_GP4	ALT2	fec	RDATA[2]	
	ALT3	—	DI2_PIN1	
	ALT4	ipu	DI2_PIN15	
	ALT6	emi	EMI_DEBUG[32]	
DISP2_DAT0	ALT0	ipu	DISP2_DAT[0]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[4]	
	ALT2	fec	RDATA[3]	
	ALT3	usboh3	USBH3_CLK	
	ALT4	kpp	COL[6]	
	ALT5	uart3	RXD_MUX	
	ALT6	emi	EMI_DEBUG[33]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP2_DAT1	ALT0	ipu	DISP2_DAT[1]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[5]	
	ALT2	fec	RX_ER	
	ALT3	usboh3	USBH3_DIR	
	ALT4	kpp	COL[7]	
	ALT5	uart3	TXD_MUX	
	ALT6	emi	EMI_DEBUG[34]	
DISP2_DAT2	No Muxing (ALT0)	ipu	DISP2_DAT[2]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP2_DAT3		ipu	DISP2_DAT[3]	
DISP2_DAT4	No Muxing (ALT0)	ipu	DISP2_DAT[4]	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
DISP2_DAT5		ipu	DISP2_DAT[5]	
DISP2_DAT6	ALT0	ipu	DISP2_DAT[6]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[6]	
	ALT2	fec	TDATA[1]	
	ALT3	usboh3	USBH3_STP	
	ALT4	kpp	ROW[4]	
	ALT5	gpio1	GPIO[19]	
	ALT6	emi	EMI_DEBUG[35]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP2_DAT7	ALT0	ipu	DISP2_DAT[7]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_HS_IN[7]	
	ALT2	fec	TDATA[2]	
	ALT3	usboh3	USBH3_NXT	
	ALT4	kpp	ROW[5]	
	ALT5	gpio1	GPIO[29]	
	ALT6	emi	EMI_DEBUG[36]	
DISP2_DAT8	ALT0	ipu	DISP2_DAT[8]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[0]	
	ALT2	fec	TDATA[3]	
	ALT3	usboh3	USBH3_DATA0	
	ALT4	kpp	ROW[6]	
	ALT5	gpio1	GPIO[30]	
	ALT6	emi	EMI_DEBUG[37]	
DISP2_DAT9	ALT0	ipu	DISP2_DAT[9]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[1]	
	ALT2	fec	TX_EN	
	ALT3	usboh3	USBH3_DATA1	
	ALT4	audmux	AUD6_RXC	
	ALT5	gpio1	GPIO[31]	
	ALT6	emi	EMI_DEBUG[38]	
DISP2_DAT10	ALT0	ipu	DISP2_DAT[10]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[2]	
	ALT2	fec	COL	
	ALT3	usboh3	USBH3_DATA2	
	ALT4	kpp	ROW[7]	
	ALT5	ipu	SER_DISP2_CS	
	ALT6	emi	EMI_DEBUG[39]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP2_DAT11	ALT0	ipu	DISP2_DAT[11]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[3]	
	ALT2	fec	RX_CLK	
	ALT3	usboh3	USBH3_DATA3	
	ALT4	audmux	AUD6_TXD	
	ALT6	emi	EMI_DEBUG[40]	
	ALT7	gpio1	GPIO[10]	
DISP2_DAT12	ALT0	ipu	DISP2_DAT[12]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[4]	
	ALT2	fec	RX_DV	
	ALT3	usboh3	USBH3_DATA4	
	ALT4	audmux	AUD6_RXD	
	ALT6	emi	EMI_DEBUG[41]	
DISP2_DAT13	ALT0	ipu	DISP2_DAT[13]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[5]	
	ALT2	fec	TX_CLK	
	ALT3	usboh3	USBH3_DATA5	
	ALT4	audmux	AUD6_TXC	
	ALT6	emi	EMI_DEBUG[42]	
DISP2_DAT14	ALT0	ipu	DISP2_DAT[14]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[6]	
	ALT2	fec	RDATA[0]	
	ALT3	usboh3	USBH3_DATA6	
	ALT4	audmux	AUD6_TXFS	
	ALT6	emi	EMI_DEBUG[43]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
DISP2_DAT15	ALT0	ipu	DISP2_DAT[15]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	—	DATA_ESC_IN[7]	
	ALT2	fec	TDATA[0]	
	ALT3	usboh3	USBH3_DATA7	
	ALT4	audmux	AUD6_RXFS	
	ALT5	ipu	SER_DISP1_CS	
	ALT6	emi	EMI_DEBUG[44]	
SD1_CMD	ALT0	esdhc1	CMD	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Enabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_RXFS	
	ALT2	cspi	MOSI	
SD1_CLK	ALT0	esdhc1	CLK	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_RXC	
	ALT2	cspi	SCLK	
SD1_DATA0	ALT0	esdhc1	DAT0	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_TXD	
	ALT2	cspi	MISO	
SD1_DATA1	ALT0	esdhc1	DAT1	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_RXD	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
SD1_DATA2	ALT0	esdhc1	DAT2	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_TXC	
SD1_DATA3	ALT0	esdhc1	DAT3	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(360 KΩ PD) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	audmux	AUD5_TXFS	
	ALT2	cspi	SS1	
GPIO1_0	ALT0	esdhc1	CD	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio1	GPIO[0]	
	ALT2	cspi	SS2	
GPIO1_1	ALT0	esdhc1	WP	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	gpio1	GPIO[1]	
	ALT2	cspi	MISO	
SD2_CMD	ALT0	esdhc2	CMD	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Enabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	i2c1	SCL	
	ALT2	cspi	MOSI	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
SD2_CLK	ALT0	esdhc2	CLK	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	i2c1	SDA	
	ALT2	cspi	SCLK	
SD2_DATA0	ALT0	esdhc2	DAT0	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	esdhc1	DAT4	
	ALT2	cspi	MISO	
SD2_DATA1	ALT0	esdhc2	DAT1	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	esdhc1	DAT5	
	ALT2	usboh3	H2_DP	
SD2_DATA2	ALT0	esdhc2	DAT2	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(47 KΩ PU) dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	esdhc1	DAT6	
	ALT2	usboh3	H2_DM	
SD2_DATA3	ALT0	esdhc2	DAT3	Drive Strength—CFG(High) low/high output voltage—CFG(High) Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(360 KΩ PD) dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) test_ts—Disabled
	ALT1	esdhc1	DAT7	
	ALT2	cspi	SS2	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
GPIO1_2	ALT0	gpio1	GPIO[2]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	pwm1	PWMO	
	ALT2	i2c2	SCL	
	ALT5	ccm	CCM_OUT_2	
	ALT6	dpllip1	TOG_EN	
	ALT7	ccm	PLL1_BYP	
GPIO1_3	ALT0	gpio1	GPIO[3]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	pwm2	PWMO	
	ALT2	i2c2	SDA	
	ALT5	ccm	CLKO2	
	ALT6	gpt	CLKIN	
	ALT7	ccm	PLL2_BYP	
RESET_IN_B	No Muxing (ALT0)	src	RESET_B	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—Enabled Slew Rate—SLOW test_ts—Disabled
POR_B		src	POR_B	
BOOT_MODE1	No Muxing (ALT0)	src	BOOT_MODE[1]	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—100 KΩ PU Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—Enabled Slew Rate—SLOW test_ts—Disabled
BOOT_MODE0		src	BOOT_MODE[0]	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
PMIC_RDY	No Muxing (ALT0)	gpc	PMIC_RDY	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—Enabled Slew Rate—SLOW test_ts—Disabled
CKIL		ccm	CKIL	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—Disabled Slew Rate—SLOW test_ts—Disabled
PMIC_STBY_REQ	No Muxing (ALT0)	ccm	PMIC_VSTBY_REQ	Drive Strength—CFG(High) Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
PMIC_ON_REQ		srtc	SRTCALARM	Drive Strength—High Hyst. Enable—Disabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—FAST test_ts—Disabled
PMIC_INT_REQ	ALT0	tzic	PWRFAIL_INT	Drive Strength—CFG(High) Hyst. Enable—CFG(Enabled) Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	tigerp_platform_ne_32k_256k	PMU_IRQ_B	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
CLK_SS	No Muxing (ALT0)	ccm	CLKSS	Drive Strength—Low Hyst. Enable—Enabled Pull/Keep Select—CFG(Pull) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—NA Pull/Keep Enable—Enabled Slew Rate—SLOW test_ts—Disabled
CKIH1		camp1	CKIH	—
GPIO1_4	ALT0	gpio1	GPIO[4]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	sdma	SDMA_EXT_EVENT[0]	
	ALT2	wdog1	WDOG_B	
	ALT3	emi	RDY	
	ALT4	ccm	DI2_EXT_CLK	
	ALT6	gpt	CAPIN1	
	ALT7	dpllip1	TOG_EN	
GPIO1_5	ALT0	gpio1	GPIO[5]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	sdma	SDMA_EXT_EVENT[1]	
	ALT2	wdog2	WDOG_B	
	ALT3	ipu	DI2_PIN16	
	ALT5	ccm	CLKO	
	ALT6	ccm	CSI2_MCLK	
GPIO1_6	ALT0	gpio1	GPIO[6]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	ccm	SSI_EXT2_CLK	
	ALT2	—	MCT_EXT_ACT_TRIG	
	ALT3	ccm	REF_EN_B	
	ALT4	ipu	DI2_PIN17	
	ALT5	epit2	EPITO	
	ALT6	gpt	CAPIN2	
	ALT7	csu	TD	

Table 4-1. i.MX51 Signal Multiplexing and I/O Configurations (continued)

Pad Name	Mode	Instance	Port	Pad Settings
GPIO1_7	ALT0	gpio1	GPIO[7]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	ccm	SSI_EXT1_CLK	
	ALT2	spdif	OUT1	
	ALT3	ccm	CCM_OUT_0	
	ALT5	epit1	EPITO	
	ALT6	esdhc2	WP	
	ALT7	dpllip1	TOG_EN	
GPIO1_8	ALT0	gpio1	GPIO[8]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PU) Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	usboh3	USB_PWR	
	ALT2	—	CSI2_DATA_EN	
	ALT4	ccm	CLKO2	
	ALT6	esdhc2	CD	
	ALT7	src	TESTER_ACK	
GPIO1_9	ALT0	gpio1	GPIO[9]	Drive Strength—CFG(High) Hyst. Enable—CFG(Disabled) Pull/Keep Select—CFG(Keep) Pull Up/Down Config.—CFG(100 KΩ PD) Strength mode—4_level dse test—regular Open Drain Enable—CFG(Disabled) Pull/Keep Enable—CFG(Enabled) Slew Rate—CFG(FAST) test_ts—Disabled
	ALT1	usboh3	USB_OC	
	ALT2	ipu	DI2_D1_CS	
	ALT3	ccm	CCM_OUT_1	
	ALT4	ccm	CLKO	
	ALT6	esdhc2	LCTL	
	ALT7	ipu	SER_DISP2_CS	
TEST_MODE	No Muxing (ALT0)	tcu	TEST_MODE	Drive Strength—Low Hyst. Enable—Disabled Pull/Keep Select—Pull Pull Up/Down Config.—100 KΩ PD Strength mode—4_level dse test—regular Open Drain Enable—Disabled Pull/Keep Enable—Enabled Slew Rate—SLOW test_ts—Disabled
CKIH2		camp2	CKIH	—

Table 4-2 lists muxing options sorted by module/function.

Table 4-2. Muxing Options Sorted by Module

Instance	Port	Pad	Mode
HSI2C	SCL	I2C1_CLK	ALT0
	SDA	I2C1_DAT	ALT0
TZIC	PWRFAIL_INT	PMIC_INT_REQ	ALT0
RTIC	RTIC_DONE_INT	KEY_ROW2	ALT1
	RTIC_SEC_VIO	KEY_ROW1	ALT1
SCC	FAIL_STATE	EIM_D22	ALT3
	RANDOM	KEY_ROW1	ALT5
	RANDOM_V	KEY_ROW0	ALT5
	SEC_STATE	EIM_D23	ALT3
SRTC	SRTCALARM	PMIC_ON_REQ	No Muxing (ALT0)
	SRTC_ALARM_DEB	EIM_D20	ALT4
		EIM_D21	ALT3
TCU	TEST_MODE	TEST_MODE	No Muxing (ALT0)
GPC	PMIC_RDY	PMIC_RDY	No Muxing (ALT0)
ECSPI1	MISO	CSPI1_MISO	ALT0
	MOSI	CSPI1_MOSI	ALT0
	RDY	CSPI1_RDY	ALT0
	SCLK	CSPI1_SCLK	ALT0
	SS0	CSPI1_SS0	ALT0
	SS1	CSPI1_SS1	ALT0
	SS2	DI1_PIN11	ALT7
	SS3	USBH1_DATA7	ALT1
ECSPI2	MISO	NANDF_RB3	ALT2
	MOSI	NANDF_D15	ALT2
	RDY	NANDF_RB1	ALT2
	SCLK	NANDF_RB2	ALT2
	SS0	NANDF_RDY_INT	ALT2
	SS1	NANDF_D12	ALT2
		NANDF_RB0	ALT5
	SS2	NANDF_D13	ALT2
	SS3	NANDF_D14	ALT2
		USBH1_DATA7	ALT5

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
UART1	CTS	UART1_CTS	ALT0
	DCD	KEY_COL5	ALT1
	DSR	UART3_TXD	ALT0
	DTR	UART3_RXD	ALT0
	RI	KEY_COL4	ALT1
	RTS	UART1_RTS	ALT0
	RXD_MUX	UART1_RXD	ALT0
	TXD_MUX	UART1_TXD	ALT0
UART2	CTS	EIM_D16	ALT3
		EIM_D25	ALT4
		USBH1_DATA0	ALT1
	RTS	EIM_D19	ALT3
		EIM_D26	ALT4
		USBH1_DATA3	ALT1
UART2	RXD_MUX	EIM_D17	ALT3
		UART2_RXD	ALT0
		USBH1_DATA1	ALT1
	TXD_MUX	EIM_D18	ALT3
		UART2_TXD	ALT0
		USBH1_DATA2	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
UART3	CTS	EIM_D17	ALT4
		EIM_D24	ALT3
		KEY_COL5	ALT2
		USBH1_CLK	ALT7
	RTS	EIM_D18	ALT4
		EIM_D27	ALT3
		KEY_COL4	ALT2
		USBH1_DIR	ALT7
	RXD_MUX	AUD3_BB_RXD	ALT1
		DISP2_DAT0	ALT5
		EIM_D25	ALT3
		UART3_RXD	ALT1
		USBH1_STP	ALT5
	TXD_MUX	AUD3_BB_FS	ALT1
		DISP2_DAT1	ALT5
		EIM_D26	ALT3
		UART3_TXD	ALT1
		USBH1_NXT	ALT5
DPLLIP1	TOG_EN	GPIO1_2	ALT6
		GPIO1_4	ALT7
		GPIO1_7	ALT7
		NANDF_RB3	ALT4
TIGERP_PLATFORM_NE_32K_256K	CTI_TRIGIN7	KEY_COL5	ALT5
	CTI_TRIGIN_ACK7	KEY_COL4	ALT5
	CTI_TRIGOUT6	OWIRE_LINE	ALT5
	CTI_TRIGOUT7	KEY_COL3	ALT5
	CTI_TRIGOUT_ACK6	DISP1_DAT22	ALT3
	CTI_TRIGOUT_ACK7	DISP1_DAT23	ALT3
	PMU_IRQ_B	PMIC_INT_REQ	ALT1
WDOG1	WDOG_B	GPIO1_4	ALT2
	WDOG_RST_B_DEB	DISPB2_SER_DIO	ALT6
WDOG2	WDOG_B	GPIO1_5	ALT2
	WDOG_RST_B_DEB	DISPB2_SER_CLK	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	DRAM_A[0]	DRAM_A0	No Muxing (ALT0)
	DRAM_A[10]	DRAM_A10	No Muxing (ALT0)
	DRAM_A[11]	DRAM_A11	No Muxing (ALT0)
	DRAM_A[12]	DRAM_A12	No Muxing (ALT0)
	DRAM_A[13]	DRAM_A13	No Muxing (ALT0)
	DRAM_A[14]	DRAM_A14	No Muxing (ALT0)
	DRAM_A[1]	DRAM_A1	No Muxing (ALT0)
	DRAM_A[2]	DRAM_A2	No Muxing (ALT0)
	DRAM_A[3]	DRAM_A3	No Muxing (ALT0)
EMI	DRAM_A[4]	DRAM_A4	No Muxing (ALT0)
	DRAM_A[5]	DRAM_A5	No Muxing (ALT0)
	DRAM_A[6]	DRAM_A6	No Muxing (ALT0)
	DRAM_A[7]	DRAM_A7	No Muxing (ALT0)
	DRAM_A[8]	DRAM_A8	No Muxing (ALT0)
	DRAM_A[9]	DRAM_A9	No Muxing (ALT0)
	DRAM_CAS	DRAM_CAS	No Muxing (ALT0)
	DRAM_CS0	DRAM_CS0	No Muxing (ALT0)
	DRAM_CS1	DRAM_CS1	ALT0
	DRAM_DQM[0]	DRAM_DQM0	No Muxing (ALT0)
	DRAM_DQM[1]	DRAM_DQM1	No Muxing (ALT0)
	DRAM_DQM[2]	DRAM_DQM2	No Muxing (ALT0)
	DRAM_DQM[3]	DRAM_DQM3	No Muxing (ALT0)
	DRAM_D[0]	DRAM_D0	No Muxing (ALT0)
	DRAM_D[10]	DRAM_D10	No Muxing (ALT0)
	DRAM_D[11]	DRAM_D11	No Muxing (ALT0)
	DRAM_D[12]	DRAM_D12	No Muxing (ALT0)
	DRAM_D[13]	DRAM_D13	No Muxing (ALT0)
	DRAM_D[14]	DRAM_D14	No Muxing (ALT0)
	DRAM_D[15]	DRAM_D15	No Muxing (ALT0)
	DRAM_D[16]	DRAM_D16	No Muxing (ALT0)
	DRAM_D[17]	DRAM_D17	No Muxing (ALT0)
	DRAM_D[18]	DRAM_D18	No Muxing (ALT0)
	DRAM_D[19]	DRAM_D19	No Muxing (ALT0)

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	DRAM_D[1]	DRAM_D1	No Muxing (ALT0)
	DRAM_D[20]	DRAM_D20	No Muxing (ALT0)
	DRAM_D[21]	DRAM_D21	No Muxing (ALT0)
	DRAM_D[22]	DRAM_D22	No Muxing (ALT0)
	DRAM_D[23]	DRAM_D23	No Muxing (ALT0)
	DRAM_D[24]	DRAM_D24	No Muxing (ALT0)
	DRAM_D[25]	DRAM_D25	No Muxing (ALT0)
	DRAM_D[26]	DRAM_D26	No Muxing (ALT0)
	DRAM_D[27]	DRAM_D27	No Muxing (ALT0)
	DRAM_D[28]	DRAM_D28	No Muxing (ALT0)
	DRAM_D[29]	DRAM_D29	No Muxing (ALT0)
	DRAM_D[2]	DRAM_D2	No Muxing (ALT0)
	DRAM_D[30]	DRAM_D30	No Muxing (ALT0)
	DRAM_D[31]	DRAM_D31	No Muxing (ALT0)
	DRAM_D[3]	DRAM_D3	No Muxing (ALT0)
	DRAM_D[4]	DRAM_D4	No Muxing (ALT0)
	DRAM_D[5]	DRAM_D5	No Muxing (ALT0)
	DRAM_D[6]	DRAM_D6	No Muxing (ALT0)
	DRAM_D[7]	DRAM_D7	No Muxing (ALT0)
	DRAM_D[8]	DRAM_D8	No Muxing (ALT0)
	DRAM_D[9]	DRAM_D9	No Muxing (ALT0)
	DRAM_ODT0	EIM_SDODT0	No Muxing (ALT0)
	DRAM_ODT1	EIM_SDODT1	No Muxing (ALT0)
	DRAM_RAS	DRAM_RAS	No Muxing (ALT0)

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	DRAM_SDBA[0]	EIM_SDBA0	No Muxing (ALT0)
	DRAM_SDBA[1]	EIM_SDBA1	No Muxing (ALT0)
	DRAM_SDBA[2]	EIM_SDBA2	No Muxing (ALT0)
	DRAM_SDCKE[0]	DRAM_SDCKE0	No Muxing (ALT0)
	DRAM_SDCKE[1]	DRAM_SDCKE1	No Muxing (ALT0)
	DRAM_SDCLK	DRAM_SDCLK	No Muxing (ALT0)
	DRAM_SDCLK_B	DRAM_SDCLK_B	No Muxing (ALT0)
	DRAM_SDQS[0]	DRAM_SDQS0	No Muxing (ALT0)
	DRAM_SDQS[1]	DRAM_SDQS1	No Muxing (ALT0)
	DRAM_SDQS[2]	DRAM_SDQS2	No Muxing (ALT0)
	DRAM_SDQS[3]	DRAM_SDQS3	No Muxing (ALT0)
	DRAM_SDQS_B[0]	DRAM_SDQS0_B	No Muxing (ALT0)
	DRAM_SDQS_B[1]	DRAM_SDQS1_B	No Muxing (ALT0)
	DRAM_SDQS_B[2]	DRAM_SDQS2_B	No Muxing (ALT0)
	DRAM_SDQS_B[3]	DRAM_SDQS3_B	No Muxing (ALT0)
	DRAM_SDWE	DRAM_SDWE	No Muxing (ALT0)
	DSTROBE	KEY_ROW0	ALT1
	EIM_A[16]	EIM_A16	ALT0
	EIM_A[17]	EIM_A17	ALT0
	EIM_A[18]	EIM_A18	ALT0
	EIM_A[19]	EIM_A19	ALT0
	EIM_A[20]	EIM_A20	ALT0
	EIM_A[21]	EIM_A21	ALT0
	EIM_A[22]	EIM_A22	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	EIM_A[23]	EIM_A23	ALT0
	EIM_A[24]	EIM_A24	ALT0
	EIM_A[25]	EIM_A25	ALT0
	EIM_A[26]	EIM_A26	ALT0
	EIM_A[27]	EIM_A27	ALT0
	EIM_BCLK	EIM_BCLK	No Muxing (ALT0)
	EIM_CRE	EIM_CRE	ALT0
	EIM_CS0	EIM_CS0	ALT0
	EIM_CS1	EIM_CS1	ALT0
	EIM_CS2	EIM_CS2	ALT0
	EIM_CS3	EIM_CS3	ALT0
	EIM_CS4	EIM_CS4	ALT0
	EIM_CS5	EIM_CS5	ALT0
	EIM_DA[0]	EIM_DA0	ALT0
	EIM_DA[10]	EIM_DA10	ALT0
	EIM_DA[11]	EIM_DA11	ALT0
	EIM_DA[12]	EIM_DA12	ALT0
	EIM_DA[13]	EIM_DA13	ALT0
	EIM_DA[14]	EIM_DA14	ALT0
	EIM_DA[15]	EIM_DA15	ALT0
	EIM_DA[1]	EIM_DA1	ALT0
	EIM_DA[2]	EIM_DA2	ALT0
	EIM_DA[3]	EIM_DA3	ALT0
	EIM_DA[4]	EIM_DA4	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	EIM_DA[5]	EIM_DA5	ALT0
	EIM_DA[6]	EIM_DA6	ALT0
	EIM_DA[7]	EIM_DA7	ALT0
	EIM_DA[8]	EIM_DA8	ALT0
	EIM_DA[9]	EIM_DA9	ALT0
	EIM_EB[0]	EIM_EB0	ALT0
	EIM_EB[1]	EIM_EB1	ALT0
	EIM_EB[2]	EIM_EB2	ALT0
	EIM_EB[3]	EIM_EB3	ALT0
	EIM_LBA	EIM_LBA	ALT0
	EIM_NFC_D[0]	NANDF_D0	ALT0
	EIM_NFC_D[10]	NANDF_D10	ALT0
	EIM_NFC_D[11]	NANDF_D11	ALT0
	EIM_NFC_D[12]	NANDF_D12	ALT0
	EIM_NFC_D[13]	NANDF_D13	ALT0
	EIM_NFC_D[14]	NANDF_D14	ALT0
	EIM_NFC_D[15]	NANDF_D15	ALT0
	EIM_NFC_D[1]	NANDF_D1	ALT0
	EIM_NFC_D[2]	NANDF_D2	ALT0
	EIM_NFC_D[3]	NANDF_D3	ALT0
	EIM_NFC_D[4]	NANDF_D4	ALT0
	EIM_NFC_D[5]	NANDF_D5	ALT0
	EIM_NFC_D[6]	NANDF_D6	ALT0
	EIM_NFC_D[7]	NANDF_D7	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	EIM_NFC_D[8]	NANDF_D8	ALT0
	EIM_NFC_D[9]	NANDF_D9	ALT0
	EIM_OE	EIM_OE	ALT0
	EIM_RW	EIM_RW	No Muxing (ALT0)
	EIM_WAIT	EIM_WAIT	No Muxing (ALT0)
	EMI_DEBUG[0]	USBH1_CLK	ALT6
	EMI_DEBUG[10]	USBH1_DATA6	ALT6
	EMI_DEBUG[11]	USBH1_DATA7	ALT6
	EMI_DEBUG[12]	UART1_RXD	ALT6
	EMI_DEBUG[13]	UART1_TXD	ALT6
	EMI_DEBUG[14]	UART1_RTS	ALT6
	EMI_DEBUG[15]	UART1_CTS	ALT6
	EMI_DEBUG[16]	UART2_RXD	ALT6
	EMI_DEBUG[17]	UART2_TXD	ALT6
	EMI_DEBUG[18]	UART3_RXD	ALT6
	EMI_DEBUG[19]	UART3_TXD	ALT6
	EMI_DEBUG[1]	USBH1_DIR	ALT6
	EMI_DEBUG[20]	KEY_ROW0	ALT6
	EMI_DEBUG[21]	KEY_ROW1	ALT6
	EMI_DEBUG[22]	KEY_ROW2	ALT6
	EMI_DEBUG[23]	KEY_ROW3	ALT6
	EMI_DEBUG[24]	DI1_PIN3	ALT6
	EMI_DEBUG[25]	DI1_PIN2	ALT6
	EMI_DEBUG[26]	DI_GP1	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	EMI_DEBUG[27]	DI_GP2	ALT6
	EMI_DEBUG[28]	DI_GP3	ALT6
	EMI_DEBUG[29]	DI2_PIN4	ALT6
	EMI_DEBUG[2]	USBH1_STP	ALT6
	EMI_DEBUG[30]	DI2_PIN2	ALT6
	EMI_DEBUG[31]	DI2_PIN3	ALT6
	EMI_DEBUG[32]	DI_GP4	ALT6
	EMI_DEBUG[33]	DISP2_DAT0	ALT6
	EMI_DEBUG[34]	DISP2_DAT1	ALT6
	EMI_DEBUG[35]	DISP2_DAT6	ALT6
	EMI_DEBUG[36]	DISP2_DAT7	ALT6
	EMI_DEBUG[37]	DISP2_DAT8	ALT6
	EMI_DEBUG[38]	DISP2_DAT9	ALT6
	EMI_DEBUG[39]	DISP2_DAT10	ALT6
	EMI_DEBUG[3]	USBH1_NXT	ALT6
	EMI_DEBUG[40]	DISP2_DAT11	ALT6
	EMI_DEBUG[41]	DISP2_DAT12	ALT6
	EMI_DEBUG[42]	DISP2_DAT13	ALT6
	EMI_DEBUG[43]	DISP2_DAT14	ALT6
	EMI_DEBUG[44]	DISP2_DAT15	ALT6
	EMI_DEBUG[45]	CSI2_D13	ALT6
	EMI_DEBUG[46]	CSI2_D18	ALT6
	EMI_DEBUG[47]	CSI2_D19	ALT6
	EMI_DEBUG[48]	CSI2_VSYNC	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	EMI_DEBUG[49]	CSI2_HSYNC	ALT6
	EMI_DEBUG[4]	USBH1_DATA0	ALT6
	EMI_DEBUG[50]	CSI2_PIXCLK	ALT6
	EMI_DEBUG[5]	USBH1_DATA1	ALT6
	EMI_DEBUG[6]	USBH1_DATA2	ALT6
	EMI_DEBUG[7]	USBH1_DATA3	ALT6
	EMI_DEBUG[8]	USBH1_DATA4	ALT6
	EMI_DEBUG[9]	USBH1_DATA5	ALT6
	NANDF_ALE	NANDF_ALE	ALT0
	NANDF_CLE	NANDF_CLE	ALT0
	NANDF_CS0	NANDF_CS0	ALT0
	NANDF_CS1	NANDF_CS1	ALT0
	NANDF_CS2	NANDF_CS2	ALT0
	NANDF_CS3	NANDF_CS3	ALT0
	NANDF_CS4	NANDF_CS4	ALT0
	NANDF_CS5	NANDF_CS5	ALT0
	NANDF_CS6	NANDF_CS6	ALT0
	NANDF_CS7	NANDF_CS7	ALT0
	NANDF_RB0	NANDF_RB0	ALT0
	NANDF_RB1	NANDF_RB1	ALT0
	NANDF_RB2	NANDF_RB2	ALT0
	NANDF_RB3	NANDF_RB3	ALT0
	NANDF_RE_B	NANDF_RE_B	ALT0
	NANDF_WE_B	NANDF_WE_B	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
EMI	NANDF_WP_B	NANDF_WP_B	ALT0
	RDY	GPIO1_4	ALT3
		NANDF_RDY_INT	ALT0
	WEIM_DTACK_B	EIM_DTACK	ALT0
	WEIM_D[16]	EIM_D16	ALT0
	WEIM_D[17]	EIM_D17	ALT0
	WEIM_D[18]	EIM_D18	ALT0
	WEIM_D[19]	EIM_D19	ALT0
	WEIM_D[20]	EIM_D20	ALT0
	WEIM_D[21]	EIM_D21	ALT0
	WEIM_D[22]	EIM_D22	ALT0
	WEIM_D[23]	EIM_D23	ALT0
	WEIM_D[24]	EIM_D24	ALT0
	WEIM_D[25]	EIM_D25	ALT0
	WEIM_D[26]	EIM_D26	ALT0
	WEIM_D[27]	EIM_D27	ALT0
	WEIM_D[28]	EIM_D28	ALT0
	WEIM_D[29]	EIM_D29	ALT0
	WEIM_D[30]	EIM_D30	ALT0
	WEIM_D[31]	EIM_D31	ALT0
IPU	DI1_D0_CS	DI1_D0_CS	ALT0
	DI1_D1_CS	DI1_D1_CS	ALT0
	DI1_DISP_CLK	DI1_DISP_CLK	No Muxing (ALT0)
	DI1_PIN11	DI1_PIN11	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
IPU	DI1_PIN12	DI1_PIN12	ALT0
	DI1_PIN13	DI1_PIN13	ALT0
	DI1_PIN14	DI1_D1_CS	ALT2
	DI1_PIN15	DI1_PIN15	No Muxing (ALT0)
	DI1_PIN16	DISPB2_SER_RS	ALT2
	DI1_PIN17	DISPB2_SER_CLK	ALT2
	DI1_PIN2	DI1_PIN2	ALT0
	DI1_PIN3	DI1_PIN3	ALT0
	DI1_PIN4	EIM_A25	ALT6
	DI1_PIN5	DI1_D1_CS	ALT3
	DI1_PIN6	DISPB2_SER_DIO	ALT3
	DI1_PIN7	DISPB2_SER_CLK	ALT3
	DI1_PIN8	DISPB2_SER_RS	ALT3
	DI2_D0_CS	DISP1_DAT22	ALT6
	DI2_D1_CS	DISP1_DAT23	ALT6
		GPIO1_9	ALT2
	DI2_DISP_CLK	DI2_DISP_CLK	ALT0
	DI2_PIN11	DISP1_DAT18	ALT5
	DI2_PIN12	DISP1_DAT19	ALT5
	DI2_PIN13	DISP1_DAT20	ALT5
	DI2_PIN14	DISP1_DAT21	ALT5
	DI2_PIN15	DI_GP4	ALT4
	DI2_PIN16	GPIO1_5	ALT3
	DI2_PIN17	GPIO1_6	ALT4

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
IPU	DI2_PIN2	DI2_PIN2	ALT0
	DI2_PIN3	DI2_PIN3	ALT0
	DI2_PIN4	DI2_PIN4	ALT0
	DI2_PIN5	DISP1_DAT18	ALT4
	DI2_PIN6	DISP1_DAT19	ALT4
	DI2_PIN7	DISP1_DAT20	ALT4
	DI2_PIN8	DISP1_DAT21	ALT4
	DISP1_DAT[0]	DISP1_DAT0	No Muxing (ALT0)
	DISP1_DAT[10]	DISP1_DAT10	ALT0
	DISP1_DAT[11]	DISP1_DAT11	ALT0
	DISP1_DAT[12]	DISP1_DAT12	ALT0
	DISP1_DAT[13]	DISP1_DAT13	ALT0
	DISP1_DAT[14]	DISP1_DAT14	ALT0
	DISP1_DAT[15]	DISP1_DAT15	ALT0
	DISP1_DAT[16]	DISP1_DAT16	ALT0
	DISP1_DAT[17]	DISP1_DAT17	ALT0
	DISP1_DAT[18]	DISP1_DAT18	ALT0
	DISP1_DAT[19]	DISP1_DAT19	ALT0
	DISP1_DAT[1]	DISP1_DAT1	No Muxing (ALT0)
	DISP1_DAT[20]	DISP1_DAT20	ALT0
	DISP1_DAT[21]	DISP1_DAT21	ALT0
	DISP1_DAT[22]	DISP1_DAT22	ALT0
	DISP1_DAT[23]	DISP1_DAT23	ALT0
	DISP1_DAT[2]	DISP1_DAT2	No Muxing (ALT0)

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
IPU	DISP1_DAT[3]	DISP1_DAT3	No Muxing (ALT0)
	DISP1_DAT[4]	DISP1_DAT4	No Muxing (ALT0)
	DISP1_DAT[5]	DISP1_DAT5	No Muxing (ALT0)
	DISP1_DAT[6]	DISP1_DAT6	ALT0
	DISP1_DAT[7]	DISP1_DAT7	ALT0
	DISP1_DAT[8]	DISP1_DAT8	ALT0
	DISP1_DAT[9]	DISP1_DAT9	ALT0
	DISP2_DAT[0]	DISP2_DAT0	ALT0
	DISP2_DAT[10]	DISP2_DAT10	ALT0
	DISP2_DAT[11]	DISP2_DAT11	ALT0
	DISP2_DAT[12]	DISP2_DAT12	ALT0
	DISP2_DAT[13]	DISP2_DAT13	ALT0
	DISP2_DAT[14]	DISP2_DAT14	ALT0
	DISP2_DAT[15]	DISP2_DAT15	ALT0
	DISP2_DAT[16]	DISP1_DAT22	ALT5
	DISP2_DAT[17]	DISP1_DAT23	ALT5
	DISP2_DAT[1]	DISP2_DAT1	ALT0
	DISP2_DAT[2]	DISP2_DAT2	No Muxing (ALT0)
	DISP2_DAT[3]	DISP2_DAT3	No Muxing (ALT0)
	DISP2_DAT[4]	DISP2_DAT4	No Muxing (ALT0)
	DISP2_DAT[5]	DISP2_DAT5	No Muxing (ALT0)
	DISP2_DAT[6]	DISP2_DAT6	ALT0
	DISP2_DAT[7]	DISP2_DAT7	ALT0
	DISP2_DAT[8]	DISP2_DAT8	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
IPU	DISP2_DAT[9]	DISP2_DAT9	ALT0
	DISPB1_SER_CLK	DI_GP2	ALT0
	DISPB1_SER_DIN	DI_GP4	ALT0
	DISPB1_SER_DIO	DI_GP3	ALT0
	DISPB1_SER_RS	DI_GP1	ALT0
	DISPB2_SER_CLK	DISPB2_SER_CLK	ALT0
	DISPB2_SER_DIN	DISPB2_SER_DIN	ALT0
	DISPB2_SER_DIO	DISPB2_SER_DIO	ALT0
	DISPB2_SER_RS	DISPB2_SER_RS	ALT0
	IPU_DIAG_BUS[0]	NANDF_D15	ALT6
	IPU_DIAG_BUS[10]	NANDF_D5	ALT6
	IPU_DIAG_BUS[11]	NANDF_D4	ALT6
	IPU_DIAG_BUS[12]	NANDF_D3	ALT6
	IPU_DIAG_BUS[13]	NANDF_D2	ALT6
	IPU_DIAG_BUS[14]	NANDF_D1	ALT6
	IPU_DIAG_BUS[15]	NANDF_D0	ALT6
	IPU_DIAG_BUS[1]	NANDF_D14	ALT6
	IPU_DIAG_BUS[2]	NANDF_D13	ALT6
	IPU_DIAG_BUS[3]	NANDF_D12	ALT6
	IPU_DIAG_BUS[4]	NANDF_D11	ALT6
	IPU_DIAG_BUS[5]	NANDF_D10	ALT6
	IPU_DIAG_BUS[6]	NANDF_D9	ALT6
	IPU_DIAG_BUS[7]	NANDF_D8	ALT6
	IPU_DIAG_BUS[8]	NANDF_D7	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
IPU	IPU_DIAG_BUS[9]	NANDF_D6	ALT6
	SER_DISP1_CS	DISP2_DAT15	ALT5
	SER_DISP2_CS	DISP1_DAT23	ALT4
		DISP2_DAT10	ALT5
		GPIO1_9	ALT7
	SISG[0]	EIM_A26	ALT4
		EIM_D28	ALT4
	SISG[1]	EIM_A27	ALT4
		EIM_D29	ALT4
	SISG[2]	EIM_D30	ALT4
		EIM_EB2	ALT4
	SISG[3]	EIM_D31	ALT4
		EIM_EB3	ALT4
	SISG[4]	EIM_CS1	ALT4
		EIM_D17	ALT5
	SISG[5]	EIM_CS2	ALT4
		EIM_D18	ALT5
	SNOOP2	KEY_COL2	ALT1
SDMA	DEBUG_BUS_DEVICE[0]	USBH1_DATA3	ALT4
	DEBUG_BUS_DEVICE[1]	USBH1_DATA4	ALT4
	DEBUG_BUS_DEVICE[2]	USBH1_DATA5	ALT4
	DEBUG_BUS_DEVICE[3]	USBH1_DATA6	ALT4
	DEBUG_BUS_DEVICE[4]	USBH1_DATA7	ALT4
	DEBUG_BUS_ERROR	CSI2_VSYNC	ALT4

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
SDMA	DEBUG_BUS_RWB	USBH1_NXT	ALT4
	DEBUG_CORE_RUN	USBH1_CLK	ALT4
	DEBUG_CORE_STATE[0]	NANDF_D1	ALT4
	DEBUG_CORE_STATE[1]	NANDF_D0	ALT4
	DEBUG_CORE_STATE[2]	USBH1_DATA0	ALT4
	DEBUG_CORE_STATE[3]	USBH1_DATA1	ALT4
	DEBUG_EVENT_CHANNEL[0]	NANDF_CS5	ALT4
	DEBUG_EVENT_CHANNEL[1]	NANDF_CS6	ALT4
	DEBUG_EVENT_CHANNEL[2]	NANDF_CS7	ALT4
	DEBUG_EVENT_CHANNEL[3]	NANDF_RDY_INT	ALT4
	DEBUG_EVENT_CHANNEL[4]	NANDF_RB0	ALT4
	DEBUG_EVENT_CHANNEL[5]	NANDF_CS1	ALT4
	DEBUG_EVENT_CHANNEL_SEL	USBH1_STP	ALT4
	DEBUG_EVT_CHN_LINES[0]	NANDF_WE_B	ALT4
	DEBUG_EVT_CHN_LINES[1]	NANDF_RE_B	ALT4
	DEBUG_EVT_CHN_LINES[2]	NANDF_ALE	ALT4
	DEBUG_EVT_CHN_LINES[3]	NANDF_CLE	ALT4
	DEBUG_EVT_CHN_LINES[4]	NANDF_WP_B	ALT4
	DEBUG_EVT_CHN_LINES[5]	NANDF_CS2	ALT4
	DEBUG_EVT_CHN_LINES[6]	NANDF_CS3	ALT4
	DEBUG_EVT_CHN_LINES[7]	NANDF_CS4	ALT4
	DEBUG_MATCHED_DMBUS	USBH1_DATA2	ALT4
	DEBUG_MODE	USBH1_DIR	ALT4
	DEBUG_PC[0]	NANDF_D15	ALT4

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
SDMA	DEBUG_PC[10]	NANDF_D5	ALT4
	DEBUG_PC[11]	NANDF_D4	ALT4
	DEBUG_PC[12]	NANDF_D3	ALT4
	DEBUG_PC[13]	NANDF_D2	ALT4
	DEBUG_PC[1]	NANDF_D14	ALT4
	DEBUG_PC[2]	NANDF_D13	ALT4
	DEBUG_PC[3]	NANDF_D12	ALT4
	DEBUG_PC[4]	NANDF_D11	ALT4
	DEBUG_PC[5]	NANDF_D10	ALT4
	DEBUG_PC[6]	NANDF_D9	ALT4
	DEBUG_PC[7]	NANDF_D8	ALT4
	DEBUG_PC[8]	NANDF_D7	ALT4
	DEBUG_PC[9]	NANDF_D6	ALT4
	DEBUG_RTBUFFER_WRITE	CSI2_D18	ALT4
	DEBUG_YIELD	CSI2_D19	ALT4
	SDMA_EXT_EVENT[0]	GPIO1_4	ALT1
	SDMA_EXT_EVENT[1]	GPIO1_5	ALT1
FEC	COL	DISP2_DAT10	ALT2
		NANDF_RB2	ALT1
	CRS	DI2_PIN4	ALT2
		EIM_CS5	ALT3
	MDC	DI2_PIN2	ALT2
		NANDF_CS3	ALT2
	MDIO	DI2_PIN3	ALT2

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
FEC	MDIO	EIM_EB2	ALT3
	RDATA[0]	DISP2_DAT14	ALT2
		NANDF_D9	ALT2
	RDATA[1]	DI2_DISP_CLK	ALT2
		EIM_EB3	ALT3
	RDATA[2]	DI_GP4	ALT2
		EIM_CS2	ALT3
	RDATA[3]	DISP2_DAT0	ALT2
		EIM_CS3	ALT3
	RX_CLK	DISP2_DAT11	ALT2
		NANDF_RB3	ALT1
	RX_DV	DISP2_DAT12	ALT2
		NANDF_D11	ALT2
	RX_ER	DISP2_DAT1	ALT2
		EIM_CS4	ALT3
	TDATA[0]	DISP2_DAT15	ALT2
		NANDF_D8	ALT2
	TDATA[1]	DISP2_DAT6	ALT2
		NANDF_CS4	ALT2
	TDATA[2]	DISP2_DAT7	ALT2
		NANDF_CS5	ALT2
	TDATA[3]	DISP2_DAT8	ALT2
		NANDF_CS6	ALT2
	TX_CLK	DISP2_DAT13	ALT2
FEC	TX_CLK	NANDF_RDY_INT	ALT1
	TX_EN	DISP2_DAT9	ALT2
		NANDF_CS7	ALT1
	TX_ER	DI_GP3	ALT2
		NANDF_CS2	ALT2

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
PATA	BUFFER_EN	NANDF_ALE	ALT1
	CS_0	NANDF_CS2	ALT1
	CS_1	NANDF_CS3	ALT1
	DA_0	NANDF_CS4	ALT1
	DA_1	NANDF_CS5	ALT1
	DA_2	NANDF_CS6	ALT1
	DIOR	NANDF_RE_B	ALT1
	DIOW	NANDF_WE_B	ALT1
	DMACK	NANDF_WP_B	ALT1
	DMARQ	NANDF_RB0	ALT1
	INTRQ	GPIO_NAND	ALT1
	IORDY	NANDF_RB1	ALT1
	PATA_DATA[0]	NANDF_D0	ALT1
	PATA_DATA[10]	NANDF_D10	ALT1
	PATA_DATA[11]	NANDF_D11	ALT1
	PATA_DATA[12]	NANDF_D12	ALT1
	PATA_DATA[13]	NANDF_D13	ALT1
	PATA_DATA[14]	NANDF_D14	ALT1
	PATA_DATA[15]	NANDF_D15	ALT1
PATA	PATA_DATA[1]	NANDF_D1	ALT1
	PATA_DATA[2]	NANDF_D2	ALT1
	PATA_DATA[3]	NANDF_D3	ALT1
	PATA_DATA[4]	NANDF_D4	ALT1
	PATA_DATA[5]	NANDF_D5	ALT1
	PATA_DATA[6]	NANDF_D6	ALT1
	PATA_DATA[7]	NANDF_D7	ALT1
	PATA_DATA[8]	NANDF_D8	ALT1
	PATA_DATA[9]	NANDF_D9	ALT1
	PATA_RESET_B	NANDF_CLE	ALT1
FIRI	RXD	UART2_TXD	ALT1
	TXD	UART2_RXD	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
CCM	CCM_OUT_0	GPIO1_7	ALT3
	CCM_OUT_1	GPIO1_9	ALT3
	CCM_OUT_2	GPIO1_2	ALT5
	CKIL	CKIL	No Muxing (ALT0)
	CLKO	DRAM_CS1	ALT1
		GPIO1_5	ALT5
		GPIO1_9	ALT4
	CLKO2	GPIO1_3	ALT5
		GPIO1_8	ALT4
	CLKSS	CLK_SS	No Muxing (ALT0)
CCM	CSI1_MCLK	CSI1_MCLK	No Muxing (ALT0)
	CSI2_MCLK	GPIO1_5	ALT6
	DI1_EXT_CLK	DI_GP1	ALT2
		EIM_CS5	ALT4
	DI2_EXT_CLK	EIM_A26	ALT6
		GPIO1_4	ALT4
	PLL1_BYP	GPIO1_2	ALT7
		KEY_COL0	ALT7
	PLL2_BYP	GPIO1_3	ALT7
		KEY_COL1	ALT7
	PLL3_BYP	KEY_COL2	ALT7
	PMIC_VSTBY_REQ	PMIC_STBY_REQ	No Muxing (ALT0)
	REF_EN_B	GPIO1_6	ALT3
	SSI_EXT1_CLK	EIM_CS4	ALT4
		GPIO1_7	ALT1
		KEY_COL5	ALT4
	SSI_EXT2_CLK	EIM_CS3	ALT4
		GPIO1_6	ALT1
		KEY_COL4	ALT4

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPT	CAPIN1	GPIO1_4	ALT6
	CAPIN2	GPIO1_6	ALT6
	CLKIN	GPIO1_3	ALT6
	CMPOUT1	EIM_D25	ALT5
		EIM_EB2	ALT7
GPT	CMPOUT2	EIM_D26	ALT5
		EIM_EB3	ALT7
		NANDF_RB1	ALT4
	CMPOUT3	NANDF_RB2	ALT4
KPP	COL[0]	KEY_COL0	ALT0
	COL[1]	KEY_COL1	ALT0
	COL[2]	KEY_COL2	ALT0
	COL[3]	KEY_COL3	ALT0
	COL[4]	KEY_COL4	ALT0
	COL[5]	KEY_COL5	ALT0
	COL[6]	DISP2_DAT0	ALT4
		EIM_D25	ALT1
	COL[7]	DISP2_DAT1	ALT4
		EIM_D26	ALT1
	ROW[0]	KEY_ROW0	ALT0
	ROW[1]	KEY_ROW1	ALT0
	ROW[2]	KEY_ROW2	ALT0
	ROW[3]	KEY_ROW3	ALT0
	ROW[4]	DISP2_DAT6	ALT4
		EIM_D28	ALT1
	ROW[5]	DISP2_DAT7	ALT4
		EIM_D29	ALT1
	ROW[6]	DISP2_DAT8	ALT4
		EIM_D30	ALT1
KPP	ROW[7]	DISP2_DAT10	ALT4
		EIM_D31	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
I2C1	SCL	CSPI1_SCLK	ALT1
		EIM_D19	ALT4
		SD2_CMD	ALT1
	SDA	CSPI1_MOSI	ALT1
		EIM_D16	ALT4
		SD2_CLK	ALT1
I2C2	SCL	EIM_D27	ALT4
		GPIO1_2	ALT2
		KEY_COL4	ALT3
		USBH1_CLK	ALT5
	SDA	EIM_D24	ALT4
		GPIO1_3	ALT2
		KEY_COL5	ALT3
		USBH1_DIR	ALT5
SPDIF	OUT1	EIM_D23	ALT4
		GPIO1_7	ALT2
		KEY_COL4	ALT6
		OWIRE_LINE	ALT6
GPIO1	GPIO[0]	GPIO1_0	ALT1
	GPIO[10]	DISP2_DAT11	ALT7
	GPIO[11]	USBH1_DATA0	ALT2
	GPIO[12]	USBH1_DATA1	ALT2

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPIO1	GPIO[13]	USBH1_DATA2	ALT2
	GPIO[14]	USBH1_DATA3	ALT2
	GPIO[15]	USBH1_DATA4	ALT2
	GPIO[16]	USBH1_DATA5	ALT2
	GPIO[17]	USBH1_DATA6	ALT2
	GPIO[18]	USBH1_DATA7	ALT2
	GPIO[19]	DISP2_DAT6	ALT5
	GPIO[1]	GPIO1_1	ALT1
	GPIO[20]	UART2_RXD	ALT3
	GPIO[21]	UART2_TXD	ALT3
	GPIO[22]	UART3_RXD	ALT3
	GPIO[23]	UART3_TXD	ALT3
	GPIO[24]	OWIRE_LINE	ALT3
	GPIO[25]	USBH1_CLK	ALT2
	GPIO[26]	USBH1_DIR	ALT2
	GPIO[27]	USBH1_STP	ALT2
	GPIO[28]	USBH1_NXT	ALT2
	GPIO[29]	DISP2_DAT7	ALT5
	GPIO[2]	GPIO1_2	ALT0
	GPIO[30]	DISP2_DAT8	ALT5
	GPIO[31]	DISP2_DAT9	ALT5
	GPIO[3]	GPIO1_3	ALT0
	GPIO[4]	GPIO1_4	ALT0
	GPIO[5]	GPIO1_5	ALT0
GPIO1	GPIO[6]	GPIO1_6	ALT0
	GPIO[7]	GPIO1_7	ALT0
	GPIO[8]	GPIO1_8	ALT0
	GPIO[9]	GPIO1_9	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPIO2	GPIO[0]	EIM_D16	ALT1
	GPIO[10]	EIM_A16	ALT1
	GPIO[11]	EIM_A17	ALT1
	GPIO[12]	EIM_A18	ALT1
	GPIO[13]	EIM_A19	ALT1
	GPIO[14]	EIM_A20	ALT1
	GPIO[15]	EIM_A21	ALT1
	GPIO[16]	EIM_A22	ALT1
	GPIO[17]	EIM_A23	ALT1
	GPIO[18]	EIM_A24	ALT1
	GPIO[19]	EIM_A25	ALT1
	GPIO[1]	EIM_D17	ALT1
	GPIO[20]	EIM_A26	ALT1
	GPIO[21]	EIM_A27	ALT1
	GPIO[22]	EIM_EB2	ALT1
	GPIO[23]	EIM_EB3	ALT1
	GPIO[24]	EIM_OE	ALT1
	GPIO[25]	EIM_CS0	ALT1
	GPIO[26]	EIM_CS1	ALT1
	GPIO[27]	EIM_CS2	ALT1
GPIO2	GPIO[28]	EIM_CS3	ALT1
	GPIO[29]	EIM_CS4	ALT1
	GPIO[2]	EIM_D18	ALT1
	GPIO[30]	EIM_CS5	ALT1
	GPIO[31]	EIM_DTACK	ALT1
	GPIO[3]	EIM_D19	ALT1
	GPIO[4]	EIM_D20	ALT1
	GPIO[5]	EIM_D21	ALT1
	GPIO[6]	EIM_D22	ALT1
	GPIO[7]	EIM_D23	ALT1
	GPIO[8]	EIM_D24	ALT1
	GPIO[9]	EIM_D27	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPIO3	GPIO[0]	DI1_PIN11	ALT4
	GPIO[10]	NANDF_RB2	ALT3
	GPIO[11]	NANDF_RB3	ALT3
	GPIO[12]	CSI1_D8	ALT3
		GPIO_NAND	ALT0
	GPIO[13]	CSI1_D9	ALT3
	GPIO[14]	CSI1_VSYNC	ALT3
	GPIO[15]	CSI1_HSYNC	ALT3
	GPIO[16]	NANDF_CS0	ALT3
	GPIO[17]	NANDF_CS1	ALT3
	GPIO[18]	NANDF_CS2	ALT3
	GPIO[19]	NANDF_CS3	ALT3

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPIO3	GPIO[1]	DI1_PIN12	ALT4
		EIM_LBA	ALT1
	GPIO[20]	NANDF_CS4	ALT3
	GPIO[21]	NANDF_CS5	ALT3
	GPIO[22]	NANDF_CS6	ALT3
	GPIO[23]	NANDF_CS7	ALT3
	GPIO[24]	NANDF_RDY_INT	ALT3
	GPIO[25]	NANDF_D15	ALT3
	GPIO[26]	NANDF_D14	ALT3
	GPIO[27]	NANDF_D13	ALT3
	GPIO[28]	NANDF_D12	ALT3
	GPIO[29]	NANDF_D11	ALT3
	GPIO[2]	DI1_PIN13	ALT4
		EIM_CRE	ALT1
	GPIO[30]	NANDF_D10	ALT3
	GPIO[31]	NANDF_D9	ALT3
	GPIO[3]	DI1_D0_CS	ALT4
		NANDF_WE_B	ALT3
	GPIO[4]	DI1_D1_CS	ALT4
		NANDF_RE_B	ALT3
	GPIO[5]	DISPB2_SER_DIN	ALT4
		NANDF_ALE	ALT3
	GPIO[6]	DISPB2_SER_DIO	ALT4
		NANDF_CLE	ALT3
GPIO3	GPIO[7]	DISPB2_SER_CLK	ALT4
		NANDF_WP_B	ALT3
	GPIO[8]	DISPB2_SER_RS	ALT4
		NANDF_RB0	ALT3
	GPIO[9]	NANDF_RB1	ALT3

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPIO4	GPIO[0]	NANDF_D8	ALT3
	GPIO[10]	CSI2_D13	ALT3
	GPIO[11]	CSI2_D18	ALT3
	GPIO[12]	CSI2_D19	ALT3
	GPIO[13]	CSI2_VSYNC	ALT3
	GPIO[14]	CSI2_HSYNC	ALT3
	GPIO[15]	CSI2_PIXCLK	ALT3
	GPIO[16]	I2C1_CLK	ALT3
	GPIO[17]	I2C1_DAT	ALT3
	GPIO[18]	AUD3_BB_TXD	ALT3
	GPIO[19]	AUD3_BB_RXD	ALT3
	GPIO[1]	NANDF_D7	ALT3
	GPIO[20]	AUD3_BB_CK	ALT3
	GPIO[21]	AUD3_BB_FS	ALT3
	GPIO[22]	CSPI1_MOSI	ALT3
	GPIO[23]	CSPI1_MISO	ALT3
	GPIO[24]	CSPI1_SS0	ALT3
	GPIO[25]	CSPI1_SS1	ALT3
	GPIO[26]	CSPI1_RDY	ALT3
GPIO4	GPIO[27]	CSPI1_SCLK	ALT3
	GPIO[28]	UART1_RXD	ALT3
	GPIO[29]	UART1_TXD	ALT3
	GPIO[2]	NANDF_D6	ALT3
	GPIO[30]	UART1_RTS	ALT3
	GPIO[31]	UART1_CTS	ALT3
	GPIO[3]	NANDF_D5	ALT3
	GPIO[4]	NANDF_D4	ALT3
	GPIO[5]	NANDF_D3	ALT3
	GPIO[6]	NANDF_D2	ALT3
	GPIO[7]	NANDF_D1	ALT3
	GPIO[8]	NANDF_D0	ALT3
	GPIO[9]	CSI2_D12	ALT3

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
SJC	DE_B	JTAG_DE_B	No Muxing (ALT0)
	DONE	KEY_ROW2	ALT5
	FAIL	KEY_ROW3	ALT5
	JTAG_ACT	KEY_COL5	ALT7
	MOD	JTAG_MOD	No Muxing (ALT0)
	TCK	JTAG_TCK	No Muxing (ALT0)
	TDI	JTAG_TDI	No Muxing (ALT0)
	TDO	JTAG_TDO	No Muxing (ALT0)
	TMS	JTAG_TMS	No Muxing (ALT0)
	TRSTB	JTAG_TRSTB	No Muxing (ALT0)
CAMP1	CKIH	CKIH1	No Muxing (ALT0)
CAMP2	CKIH	CKIH2	No Muxing (ALT0)
CSPI	MISO	GPIO1_1	ALT2
		SD1_DATA0	ALT2
		SD2_DATA0	ALT2
		USBH1_NXT	ALT1
	MOSI	NANDF_RB1	ALT6
		SD1_CMD	ALT2
		SD2_CMD	ALT2
		USBH1_DIR	ALT1
	RDY	USBH1_STP	ALT1
	SCLK	NANDF_CS2	ALT6
		SD1_CLK	ALT2
		SD2_CLK	ALT2
		USBH1_CLK	ALT1
	SS0	USBH1_DATA4	ALT1
	SS1	SD1_DATA3	ALT2
		USBH1_DATA5	ALT1
	SS2	GPIO1_0	ALT2
		SD2_DATA3	ALT2
	SS3	NANDF_CS6	ALT7
		USBH1_DATA6	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
SRC	ANY_PU_RST	KEY_COL4	ALT7
	BOOT_MODE[0]	BOOT_MODE0	No Muxing (ALT0)
	BOOT_MODE[1]	BOOT_MODE1	No Muxing (ALT0)
SRC	BT_BUS_WIDTH	DISP1_DAT15	ALT7
	BT_EEPROM_CFG	DISP1_DAT7	ALT7
	BT_HPN_EN	EIM_A23	ALT7
	BT_LPB[0]	EIM_A18	ALT7
	BT_LPB[1]	EIM_A19	ALT7
	BT_LPB_FREQ[0]	DISP1_DAT22	ALT7
	BT_LPB_FREQ[1]	DISP1_DAT23	ALT7
	BT_LPB_FREQ[2]	DISP1_DAT11	ALT7
	BT_MEM_CTL[0]	DISP1_DAT13	ALT7
	BT_MEM_CTL[1]	DISP1_DAT14	ALT7
	BT_MEM_TYPE[0]	DISP1_DAT20	ALT7
	BT_MEM_TYPE[1]	DISP1_DAT21	ALT7
	BT_MLC_SEL	DISP1_DAT12	ALT7
	BT_PAGE_SIZE[0]	DISP1_DAT16	ALT7
	BT_PAGE_SIZE[1]	DISP1_DAT17	ALT7
	BT_SPARE_SIZE	DISP1_DAT10	ALT7
	BT_SRC[0]	DISP1_DAT8	ALT7
	BT_SRC[1]	DISP1_DAT9	ALT7
	BT_UART_SRC[0]	EIM_A20	ALT7
	BT_UART_SRC[1]	EIM_A21	ALT7
	BT_USB_SRC	DISP1_DAT6	ALT7
	BT_WEIM_MUXED[0]	DISP1_DAT18	ALT7
	BT_WEIM_MUXED[1]	DISP1_DAT19	ALT7
	INT_BOOT	KEY_COL3	ALT7
SRC	OSC_FREQ_SEL[0]	EIM_A16	ALT7
	OSC_FREQ_SEL[1]	EIM_A17	ALT7
	POR_B	POR_B	No Muxing (ALT0)
	RESET_B	RESET_IN_B	No Muxing (ALT0)
	SYSTEM_RST	OWIRE_LINE	ALT7
	TESTER_ACK	GPIO1_8	ALT7

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
ELVIS_OBSERVE_MUX	OBSRV_INT_OUT0	CSI2_D18	ALT2
		EIM_D28	ALT3
	OBSRV_INT_OUT1	CSI2_D19	ALT2
		EIM_D29	ALT3
	OBSRV_INT_OUT2	CSI2_VSYNC	ALT2
		EIM_D30	ALT3
	OBSRV_INT_OUT3	CSI2_HSYNC	ALT2
		EIM_D31	ALT3
	OBSRV_INT_OUT4	CSI2_PIXCLK	ALT2
		EIM_A27	ALT3
SIM	CLK0	NANDF_CS4	ALT6
	PD0	NANDF_CS3	ALT6
	RST0	NANDF_CS5	ALT6
	RX0	NANDF_RDY_INT	ALT6
	SIM_RCV_CLK_TEST	DI1_PIN2	ALT7
	SIM_TX_CLK_TEST	DI1_PIN3	ALT7
	TX0	NANDF_CS7	ALT6
	VEN0	NANDF_CS6	ALT6
TPIU	TRACE[0]	EIM_D16	ALT6
	TRACE[10]	EIM_D26	ALT6
	TRACE[11]	EIM_D27	ALT6
	TRACE[12]	EIM_D28	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
TPIU	TRACE[13]	EIM_D29	ALT6
	TRACE[14]	EIM_D30	ALT6
	TRACE[15]	EIM_D31	ALT6
	TRACE[16]	EIM_DA0	ALT1
	TRACE[17]	EIM_DA1	ALT1
	TRACE[18]	EIM_DA2	ALT1
	TRACE[19]	EIM_DA3	ALT1
	TRACE[1]	EIM_D17	ALT6
	TRACE[20]	EIM_DA4	ALT1
	TRACE[21]	EIM_DA5	ALT1
	TRACE[22]	EIM_DA6	ALT1
	TRACE[23]	EIM_DA7	ALT1
	TRACE[24]	EIM_DA8	ALT1
	TRACE[25]	EIM_DA9	ALT1
	TRACE[26]	EIM_DA10	ALT1
	TRACE[27]	EIM_DA11	ALT1
	TRACE[28]	EIM_DA12	ALT1
	TRACE[29]	EIM_DA13	ALT1
	TRACE[2]	EIM_D18	ALT6
	TRACE[30]	EIM_DA14	ALT1
	TRACE[31]	EIM_DA15	ALT1
	TRACE[3]	EIM_D19	ALT6
	TRACE[4]	EIM_D20	ALT6
	TRACE[5]	EIM_D21	ALT6
TPIU	TRACE[6]	EIM_D22	ALT6
	TRACE[7]	EIM_D23	ALT6
	TRACE[8]	EIM_D24	ALT6
	TRACE[9]	EIM_D25	ALT6
	TRCLK	EIM_EB3	ALT2
	TRCTL	EIM_EB2	ALT2

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
GPU3D	GPU_DEBUG_OUT[0]	NANDF_D15	ALT7
	GPU_DEBUG_OUT[10]	NANDF_D5	ALT7
	GPU_DEBUG_OUT[11]	NANDF_D4	ALT7
	GPU_DEBUG_OUT[12]	NANDF_D3	ALT7
	GPU_DEBUG_OUT[13]	NANDF_D2	ALT7
	GPU_DEBUG_OUT[14]	NANDF_D1	ALT7
	GPU_DEBUG_OUT[15]	NANDF_D0	ALT7
	GPU_DEBUG_OUT[1]	NANDF_D14	ALT7
	GPU_DEBUG_OUT[2]	NANDF_D13	ALT7
	GPU_DEBUG_OUT[3]	NANDF_D12	ALT7
	GPU_DEBUG_OUT[4]	NANDF_D11	ALT7
	GPU_DEBUG_OUT[5]	NANDF_D10	ALT7
	GPU_DEBUG_OUT[6]	NANDF_D9	ALT7
	GPU_DEBUG_OUT[7]	NANDF_D8	ALT7
	GPU_DEBUG_OUT[8]	NANDF_D7	ALT7
	GPU_DEBUG_OUT[9]	NANDF_D6	ALT7
PWM1	PWMO	GPIO1_2	ALT1
PWM2		GPIO1_3	ALT1
PWM2	PWMO	UART1_TXD	ALT1
OWIRE	LINE	OWIRE_LINE	ALT0
EPIT1	EPITO	GPIO1_7	ALT5
EPIT2		GPIO1_6	ALT5

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
USBPHY	AVALID	EIM_D26	ALT7
	BISTOK	USBH1_STP	ALT7
	BVALID	EIM_D27	ALT7
	DATAOUT[0]	AUD3_BB_TXD	ALT7
	DATAOUT[10]	UART1_RXD	ALT7
	DATAOUT[11]	UART1_TXD	ALT7
	DATAOUT[12]	UART1_RTS	ALT7
	DATAOUT[13]	UART1_CTS	ALT7
	DATAOUT[14]	UART2_RXD	ALT7
	DATAOUT[15]	UART2_TXD	ALT7
	DATAOUT[1]	AUD3_BB_RXD	ALT7
	DATAOUT[2]	AUD3_BB_CK	ALT7
	DATAOUT[3]	AUD3_BB_FS	ALT7
	DATAOUT[4]	CSPI1_MOSI	ALT7
	DATAOUT[5]	CSPI1_MISO	ALT7
	DATAOUT[6]	CSPI1_SS0	ALT7
	DATAOUT[7]	CSPI1_SS1	ALT7
	DATAOUT[8]	CSPI1_RDY	ALT7

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
USBPHY	DATAOUT[9]	CSPI1_SCLK	ALT7
	ENDSESSION	EIM_D28	ALT7
	HOSTDISCONNECT	EIM_D30	ALT7
	IDDIG	EIM_D29	ALT7
	LINESTATE[0]	UART3_RXD	ALT7
	LINESTATE[1]	UART3_TXD	ALT7
	ONBIST	USBH1_NXT	ALT7
	RXACTIVE	EIM_D22	ALT7
	RXERROR	EIM_D23	ALT7
	RXVALID	EIM_D21	ALT7
	SIECLOCK	EIM_D24	ALT7
	TXREADY	EIM_D20	ALT7
	VBUSVALID	EIM_D25	ALT7
	VSTATUS[0]	USBH1_DATA0	ALT7
	VSTATUS[1]	USBH1_DATA1	ALT7
	VSTATUS[2]	USBH1_DATA2	ALT7
	VSTATUS[3]	USBH1_DATA3	ALT7
	VSTATUS[4]	USBH1_DATA4	ALT7
	VSTATUS[5]	USBH1_DATA5	ALT7
	VSTATUS[6]	USBH1_DATA6	ALT7
	VSTATUS[7]	USBH1_DATA7	ALT7
CSU	CSU_ALARM_AUT[0]	KEY_ROW3	ALT1
	CSU_ALARM_AUT[1]	KEY_COL0	ALT1
	CSU_ALARM_AUT[2]	KEY_COL1	ALT1
CSU	CSU_INT_DEB	EIM_D20	ALT3
	TD	GPIO1_6	ALT7

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
AUDMUX	AUD3_RXD	AUD3_BB_RXD	ALT0
	AUD3_TXC	AUD3_BB_CK	ALT0
	AUD3_TXD	AUD3_BB_TXD	ALT0
	AUD3_TXFS	AUD3_BB_FS	ALT0
	AUD4_RXC	EIM_D19	ALT5
	AUD4_RXD	CSPI1_MISO	ALT1
		EIM_D21	ALT5
	AUD4_RXFS	EIM_D16	ALT5
	AUD4_TXC	CSPI1_SS0	ALT1
		EIM_D22	ALT5
	AUD4_TXD	CSPI1_SS1	ALT1
		EIM_D20	ALT5
	AUD4_TXFS	CSPI1_RDY	ALT1
		EIM_D23	ALT5
	AUD5_RXC	EIM_EB3	ALT6
		SD1_CLK	ALT1
	AUD5_RXD	EIM_CS3	ALT6
		EIM_D17	ALT7
		SD1_DATA1	ALT1
	AUD5_RXFS	EIM_EB2	ALT6
		SD1_CMD	ALT1
	AUD5_TXC	EIM_CS4	ALT6

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
AUDMUX	AUD5_TXC	EIM_D18	ALT7
		SD1_DATA2	ALT1
	AUD5_TXD	EIM_CS2	ALT6
		EIM_D16	ALT7
		SD1_DATA0	ALT1
	AUD5_TXFS	EIM_CS5	ALT6
		EIM_D19	ALT7
		SD1_DATA3	ALT1
	AUD6_RXC	DISP2_DAT9	ALT4
		EIM_D27	ALT5
	AUD6_RXD	DISP2_DAT12	ALT4
		EIM_D29	ALT5
	AUD6_RXFS	DISP2_DAT15	ALT4
		EIM_D24	ALT5
	AUD6_TXC	DISP2_DAT13	ALT4
		EIM_D30	ALT5
	AUD6_TXD	DISP2_DAT11	ALT4
		EIM_D28	ALT5
	AUD6_TXFS	DISP2_DAT14	ALT4
		EIM_D31	ALT5
ESDHC1	CD	GPIO1_0	ALT0
	CLK	SD1_CLK	ALT0
ESDHC1	CMD	SD1_CMD	ALT0
	DAT0	SD1_DATA0	ALT0
	DAT1	SD1_DATA1	ALT0
	DAT2	SD1_DATA2	ALT0
	DAT3	SD1_DATA3	ALT0
	DAT4	SD2_DATA0	ALT1
	DAT5	SD2_DATA1	ALT1
	DAT6	SD2_DATA2	ALT1
	DAT7	SD2_DATA3	ALT1
	WP	GPIO1_1	ALT0

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
ESDHC2	CD	GPIO1_8	ALT6
	CLK	SD2_CLK	ALT0
	CMD	SD2_CMD	ALT0
	DAT0	SD2_DATA0	ALT0
	DAT1	SD2_DATA1	ALT0
	DAT2	SD2_DATA2	ALT0
	DAT3	SD2_DATA3	ALT0
	LCTL	GPIO1_9	ALT6
	WP	GPIO1_7	ALT6
ESDHC3	CLK	NANDF_CS7	ALT5
	CMD	NANDF_RDY_INT	ALT5
	DAT0	NANDF_D8	ALT5
		NANDF_WE_B	ALT2
	DAT1	NANDF_D9	ALT5
ESDHC3	DAT1	NANDF_RE_B	ALT2
	DAT2	NANDF_D10	ALT5
		NANDF_WP_B	ALT2
	DAT3	NANDF_D11	ALT5
		NANDF_RB0	ALT2
	DAT4	NANDF_D12	ALT5
	DAT5	NANDF_D13	ALT5
	DAT6	NANDF_D14	ALT5
	DAT7	NANDF_D15	ALT5

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
ESDHC4	CD	NANDF_D4	ALT2
	CLK	NANDF_CS2	ALT5
	CMD	NANDF_RB1	ALT5
	DAT0	NANDF_CS3	ALT5
	DAT1	NANDF_CS4	ALT5
	DAT2	NANDF_CS5	ALT5
	DAT3	NANDF_CS6	ALT5
	DAT4	NANDF_D3	ALT2
	DAT5	NANDF_D2	ALT2
	DAT6	NANDF_D1	ALT2
	DAT7	NANDF_D0	ALT2
	LCTL	NANDF_D6	ALT2
	WP	NANDF_D5	ALT2
SLM	CLK	AUD3_BB_CK	ALT1
	DATA	AUD3_BB_TXD	ALT1

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
USBOH3	H1_DM	NANDF_CS3	ALT7
	H1_DP	NANDF_CS2	ALT7
	H2_DM	SD2_DATA2	ALT2
	H2_DP	SD2_DATA1	ALT2
	H3_DM	NANDF_RB3	ALT7
	H3_DP	NANDF_RB2	ALT7
	USBH1_CLK	USBH1_CLK	ALT0
	USBH1_DATA0	USBH1_DATA0	ALT0
	USBH1_DATA1	USBH1_DATA1	ALT0
	USBH1_DATA2	USBH1_DATA2	ALT0
	USBH1_DATA3	USBH1_DATA3	ALT0
	USBH1_DATA4	USBH1_DATA4	ALT0
	USBH1_DATA5	USBH1_DATA5	ALT0
	USBH1_DATA6	USBH1_DATA6	ALT0
	USBH1_DATA7	USBH1_DATA7	ALT0
	USBH1_DIR	USBH1_DIR	ALT0
	USBH1_NXT	USBH1_NXT	ALT0
	USBH1_STP	USBH1_STP	ALT0
	USBH2_CLK	EIM_A24	ALT2
	USBH2_DATA0	EIM_D16	ALT2
	USBH2_DATA1	EIM_D17	ALT2
	USBH2_DATA2	EIM_D18	ALT2
	USBH2_DATA3	EIM_D19	ALT2
	USBH2_DATA4	EIM_D20	ALT2

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
USBOH3	USBH2_DATA5	EIM_D21	ALT2
	USBH2_DATA6	EIM_D22	ALT2
	USBH2_DATA7	EIM_D23	ALT2
	USBH2_DIR	EIM_A25	ALT2
	USBH2_NXT	EIM_A27	ALT2
	USBH2_STP	EIM_A26	ALT2
	USBH3_CLK	DISP2_DAT0	ALT3
		NANDF_RB3	ALT6
	USBH3_DATA0	DISP2_DAT8	ALT3
		NANDF_D7	ALT5
	USBH3_DATA1	DISP2_DAT9	ALT3
		NANDF_D6	ALT5
	USBH3_DATA2	DISP2_DAT10	ALT3
		NANDF_D5	ALT5
	USBH3_DATA3	DISP2_DAT11	ALT3
		NANDF_D4	ALT5
	USBH3_DATA4	DISP2_DAT12	ALT3
		NANDF_D3	ALT5
	USBH3_DATA5	DISP2_DAT13	ALT3
		NANDF_D2	ALT5
	USBH3_DATA6	DISP2_DAT14	ALT3
		NANDF_D1	ALT5
	USBH3_DATA7	DISP2_DAT15	ALT3
		NANDF_D0	ALT5

Table 4-2. Muxing Options Sorted by Module (continued)

Instance	Port	Pad	Mode
USBOH3	USBH3_DIR	DISP2_DAT1	ALT3
		NANDF_CS5	ALT7
	USBH3_NXT	DISP2_DAT7	ALT3
		NANDF_RB2	ALT6
	USBH3_STP	DISP2_DAT6	ALT3
		NANDF_CS4	ALT7
	USBOTG_CLK	EIM_CS4	ALT2
	USBOTG_DATA0	EIM_D24	ALT2
	USBOTG_DATA1	EIM_D25	ALT2
	USBOTG_DATA2	EIM_D26	ALT2
	USBOTG_DATA3	EIM_D27	ALT2
	USBOTG_DATA4	EIM_D28	ALT2
	USBOTG_DATA5	EIM_D29	ALT2
	USBOTG_DATA6	EIM_D30	ALT2
	USBOTG_DATA7	EIM_D31	ALT2
	USBOTG_DIR	EIM_CS5	ALT2
	USBOTG_NXT	EIM_CS3	ALT2
	USBOTG_STP	EIM_CS2	ALT2
	USB_OC	GPIO1_9	ALT1
	USB_PWR	GPIO1_8	ALT1

4.1.2 Daisy Chaining Settings

Table 4-3 shows the daisy chain settings.

Table 4-3. i.MX51 Daisy Chain Settings

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
AUDMUX	p4_input_da_amx	Module: AUDMUX, Protocol: AUD4_4W, Port: AUD4_RXD	EIM_D21	ALT5
			CSPI1_MISO	ALT1
	p4_input_db_amx	Module: AUDMUX, Protocol: AUD4_4W, Port: AUD4_TXD	EIM_D20	ALT5
			CSPI1_SS1	ALT1
	p4_input_txclk_amx	Module: AUDMUX, Protocol: AUD4_4W, Port: AUD4_TXC	EIM_D22	ALT5
			CSPI1_SS0	ALT1
	p4_input_txfs_amx	Module: AUDMUX, Protocol: AUD4_4W, Port: AUD4_TXFS	EIM_D23	ALT5
			CSPI1_RDY	ALT1
AUDMUX	p5_input_da_amx	Module: AUDMUX, Protocol: AUD5_4W, Port: AUD5_RXD	EIM_D17	ALT7
			EIM_CS3	ALT6
			SD1_DATA1	ALT1
	p5_input_db_amx	Module: AUDMUX, Protocol: AUD5_4W, Port: AUD5_TXD	EIM_D16	ALT7
			EIM_CS2	ALT6
			SD1_DATA0	ALT1
	p5_input_rxclk_amx	Module: AUDMUX, Protocol: AUD5_6W, Port: AUD5_RXC	EIM_EB3	ALT6
			SD1_CLK	ALT1
AUDMUX	p5_input_rxfs_amx	Module: AUDMUX, Protocol: AUD5_6W, Port: AUD5_RXFS	EIM_EB2	ALT6
			SD1_CMD	ALT1
	p5_input_txclk_amx	Module: AUDMUX, Protocol: AUD5_4W, Port: AUD5_TXC	EIM_D18	ALT7
			EIM_CS4	ALT6
			SD1_DATA2	ALT1
	p5_input_txfs_amx	Module: AUDMUX, Protocol: AUD5_4W, Port: AUD5_TXFS	EIM_D19	ALT7
			EIM_CS5	ALT6
			SD1_DATA3	ALT1

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
AUDMUX	p6_input_da_amx	Module: AUDMUX, Protocol: AUD6_4W, Port: AUD6_RXD	EIM_D29	ALT5
			DISP2_DAT12	ALT4
	p6_input_db_amx	Module: AUDMUX, Protocol: AUD6_4W, Port: AUD6_TXD	EIM_D28	ALT5
			DISP2_DAT11	ALT4
	p6_input_rxclk_amx	Module: AUDMUX, Protocol: AUD6_6W, Port: AUD6_RXC	EIM_D27	ALT5
			DISP2_DAT9	ALT4
	p6_input_rxfs_amx	Module: AUDMUX, Protocol: AUD6_6W, Port: AUD6_RXFS	EIM_D24	ALT5
			DISP2_DAT15	ALT4
AUDMUX	p6_input_txclk_amx	Module: AUDMUX, Protocol: AUD6_4W, Port: AUD6_TXC	EIM_D30	ALT5
			DISP2_DAT13	ALT4
	p6_input_txfs_amx	Module: AUDMUX, Protocol: AUD6_4W, Port: AUD6_TXFS	EIM_D31	ALT5
			DISP2_DAT14	ALT4
CCM	ipp_di0_clk	Module: CCM, Protocol: DISP1_SLAVE, Port: DI1_EXT_CLK	EIM_CS5	ALT4
			DI_GP1	ALT2
	ipp_di1_clk	Module: CCM, Protocol: DISP2_SLAVE, Port: DI2_EXT_CLK	EIM_A26	ALT6
			GPIO1_4	ALT4
CCM	pll1_bypass_clk	Module: CCM, Protocol: PLL1_BYP, Port: PLL1_BYP	KEY_COL0	ALT7
			GPIO1_2	ALT7
	pll2_bypass_clk	Module: CCM, Protocol: PLL2_BYP, Port: PLL2_BYP	KEY_COL1	ALT7
			GPIO1_3	ALT7
CSPI	ipp_cspi_clk_in	Module: CSPI, Protocol: MASTER, Port: SCLK	NANDF_CS2	ALT6
			USBH1_CLK	ALT1
			SD1_CLK	ALT2
			SD2_CLK	ALT2

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
CSPI	ipp_ind_miso	Module: CSPI, Protocol: MASTER, Port: MISO	USBH1_NXT	ALT1
			SD1_DATA0	ALT2
			GPIO1_1	ALT2
			SD2_DATA0	ALT2
	ipp_ind_mosi	Module: CSPI, Protocol: MASTER, Port: MOSI	NANDF_RB1	ALT6
			USBH1_DIR	ALT1
			SD1_CMD	ALT2
			SD2_CMD	ALT2
CSPI	ipp_ind_ss1_b	Module: CSPI, Protocol: MASTER, Port: SS1	USBH1_DATA5	ALT1
			SD1_DATA3	ALT2
	ipp_ind_ss2_b	Module: CSPI, Protocol: MASTER, Port: SS2	GPIO1_0	ALT2
			SD2_DATA3	ALT2
	ipp_ind_ss3_b	Module: CSPI, Protocol: MASTER, Port: SS3	NANDF_CS6	ALT7
			USBH1_DATA6	ALT1
DPLLIP1	l1t_tog_en	Module: DPLLIP, Protocol: DESENSE, Port: TOG_EN	NANDF_RB3	ALT4
			GPIO1_2	ALT6
DPLLIP1	l1t_tog_en	Module: DPLLIP, Protocol: DESENSE, Port: TOG_EN	GPIO1_4	ALT7
			GPIO1_7	ALT7
ECSPI2	ipp_ind_ss_b[1]	Module: ECSPI, Protocol: MASTER, Port: SS1	NANDF_RB0	ALT5
			NANDF_D12	ALT2
	ipp_ind_ss_b[3]	Module: ECSPI, Protocol: MASTER, Port: SS3	NANDF_D14	ALT2
			USBH1_DATA7	ALT5
EMI	ipp_ind_rdy_int	Module: EMI, Protocol: RDY, Port: RDY	NANDF_RDY_INT	ALT0
			GPIO1_4	ALT3

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
ESDHC3	ipp_dat0_in	Module: ESDHCV2, Protocol: CE_ATA, Port: DAT0	NANDF_WE_B	ALT2
			NANDF_D8	ALT5
	ipp_dat1_in	Module: ESDHCV2, Protocol: CE_ATA, Port: DAT1	NANDF_RE_B	ALT2
			NANDF_D9	ALT5
	ipp_dat2_in	Module: ESDHCV2, Protocol: CE_ATA, Port: DAT2	NANDF_WP_B	ALT2
			NANDF_D10	ALT5
	ipp_dat3_in	Module: ESDHCV2, Protocol: CE_ATA, Port: DAT3	NANDF_RB0	ALT2
			NANDF_D11	ALT5
FEC	fec_col	Module: FEC, Protocol: MII, Port: COL	NANDF_RB2	ALT1
			DISP2_DAT10	ALT2
	fec_crs	Module: FEC, Protocol: MII, Port: CRS	EIM_CS5	ALT3
			DI2_PIN4	ALT2
	fec_mdi	Module: FEC, Protocol: MII, Port: MDIO	EIM_EB2	ALT3
			DI2_PIN3	ALT2
	fec_rdata[0]	Module: FEC, Protocol: MII, Port: RDATA[0]	NANDF_D9	ALT2
			DISP2_DAT14	ALT2
FEC	fec_rdata[1]	Module: FEC, Protocol: MII, Port: RDATA[1]	EIM_EB3	ALT3
			DI2_DISP_CLK	ALT2
	fec_rdata[2]	Module: FEC, Protocol: MII, Port: RDATA[2]	EIM_CS2	ALT3
			DI_GP4	ALT2
	fec_rdata[3]	Module: FEC, Protocol: MII, Port: RDATA[3]	EIM_CS3	ALT3
			DISP2_DAT0	ALT2
	fec_rx_clk	Module: FEC, Protocol: MII, Port: RX_CLK	NANDF_RB3	ALT1
			DISP2_DAT11	ALT2

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
FEC	fec_rx_dv	Module: FEC, Protocol: MII, Port: RX_DV	NANDF_D11	ALT2
			DISP2_DAT12	ALT2
	fec_rx_er	Module: FEC, Protocol: MII, Port: RX_ER	EIM_CS4	ALT3
			DISP2_DAT1	ALT2
	fec_tx_clk	Module: FEC, Protocol: MII, Port: TX_CLK	NANDF_RDY_INT	ALT1
			DISP2_DAT13	ALT2
GPIO3	ipp_ind_g_in[1]	Module: GPIO, Protocol: GPIO[1], Port: GPIO[1]	EIM_LBA	ALT1
			DI1_PIN12	ALT4
GPIO3	ipp_ind_g_in[2]	Module: GPIO, Protocol: GPIO[2], Port: GPIO[2]	EIM_CRE	ALT1
			DI1_PIN13	ALT4
	ipp_ind_g_in[3]	Module: GPIO, Protocol: GPIO[3], Port: GPIO[3]	NANDF_WE_B	ALT3
			DI1_D0_CS	ALT4
	ipp_ind_g_in[4]	Module: GPIO, Protocol: GPIO[4], Port: GPIO[4]	NANDF_RE_B	ALT3
			DI1_D1_CS	ALT4
	ipp_ind_g_in[5]	Module: GPIO, Protocol: GPIO[5], Port: GPIO[5]	NANDF_ALE	ALT3
			DISPB2_SER_DIN	ALT4
GPIO3	ipp_ind_g_in[6]	Module: GPIO, Protocol: GPIO[6], Port: GPIO[6]	NANDF_CLE	ALT3
			DISPB2_SER_DIO	ALT4
	ipp_ind_g_in[7]	Module: GPIO, Protocol: GPIO[7], Port: GPIO[7]	NANDF_WP_B	ALT3
			DISPB2_SER_CLK	ALT4
	ipp_ind_g_in[8]	Module: GPIO, Protocol: GPIO[8], Port: GPIO[8]	NANDF_RB0	ALT3
			DISPB2_SER_RS	ALT4
	ipp_ind_g_in[12]	Module: GPIO, Protocol: GPIO[12], Port: GPIO[12]	GPIO_NAND	ALT0
			CSI1_D8	ALT3
I2C1	ipp_scl_in	Module: I2C, Protocol: I2C, Port: SCL	EIM_D19	ALT4
			CSPI1_SCLK	ALT1
			SD2_CMD	ALT1
	ipp_sda_in	Module: I2C, Protocol: I2C, Port: SDA	EIM_D16	ALT4
			CSPI1_MOSI	ALT1

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
I2C1	ipp_sda_in	Module: I2C, Protocol: I2C, Port: SDA	SD2_CLK	ALT1
I2C2	ipp_scl_in	Module: I2C, Protocol: I2C, Port: SCL	EIM_D27	ALT4
			KEY_COL4	ALT3
			USBH1_CLK	ALT5
			GPIO1_2	ALT2
	ipp_sda_in	Module: I2C, Protocol: I2C, Port: SDA	EIM_D24	ALT4
			KEY_COL5	ALT3
			USBH1_DIR	ALT5
I2C2	ipp_sda_in	Module: I2C, Protocol: I2C, Port: SDA	GPIO1_3	ALT2
IPU	ipp_di_0_ind_dispb_sd_d	Module: IPUV3E, Protocol: SER_DISP1_BIDIR, Port: DISPB1_SER_DIO	DI_GP3	ALT0
		Module: IPUV3E, Protocol: SER_DISP1_RS_UNIDIR, Port: DISPB1_SER_DIN	DI_GP4	ALT0
	ipp_di_1_ind_dispb_sd_d	Module: IPUV3E, Protocol: SER_DISP2_RS_UNIDIR, Port: DISPB2_SER_DIN	DISPB2_SER_DIN	ALT0
		Module: IPUV3E, Protocol: SER_DISP2_BIDIR, Port: DISPB2_SER_DIO	DISPB2_SER_DIO	ALT0
KPP	ipp_ind_col[6]	Module: KPP, Protocol: KPP, Port: COL[6]	EIM_D25	ALT1
			DISP2_DAT0	ALT4
	ipp_ind_col[7]	Module: KPP, Protocol: KPP, Port: COL[7]	EIM_D26	ALT1

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
KPP	ipp_ind_col[7]	Module: KPP, Protocol: KPP, Port: COL[7]	DISP2_DAT1	ALT4
	ipp_ind_row[4]	Module: KPP, Protocol: KPP, Port: ROW[4]	EIM_D28	ALT1
			DISP2_DAT6	ALT4
	ipp_ind_row[5]	Module: KPP, Protocol: KPP, Port: ROW[5]	EIM_D29	ALT1
			DISP2_DAT7	ALT4
	ipp_ind_row[6]	Module: KPP, Protocol: KPP, Port: ROW[6]	EIM_D30	ALT1
			DISP2_DAT8	ALT4
	ipp_ind_row[7]	Module: KPP, Protocol: KPP, Port: ROW[7]	EIM_D31	ALT1
KPP	ipp_ind_row[7]	Module: KPP, Protocol: KPP, Port: ROW[7]	DISP2_DAT10	ALT4
UART1	ipp_uart_rts_b	Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	UART1_RTS	ALT0
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	UART1_CTS	ALT0
	ipp_uart_rxd_mux	Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	UART1_RXD	ALT0
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	UART1_TXD	ALT0
UART2	ipp_uart_rts_b	Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	EIM_D16	ALT3
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	EIM_D19	ALT3
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	EIM_D25	ALT4

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
UART2	ipp_uart_rts_b	Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	EIM_D26	ALT4
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	USBH1_DATA0	ALT1
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	USBH1_DATA3	ALT1
	ipp_uart_rxd_mux	Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	EIM_D17	ALT3
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	EIM_D18	ALT3
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	UART2_RXD	ALT0
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	UART2_TXD	ALT0
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	USBH1_DATA1	ALT1
UART2	ipp_uart_rxd_mux	Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	USBH1_DATA2	ALT1
UART3	ipp_uart_rts_b	Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	EIM_D17	ALT4
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	EIM_D18	ALT4
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	EIM_D24	ALT3
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	EIM_D27	ALT3
			KEY_COL4	ALT2
		Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: CTS	KEY_COL5	ALT2
			USBH1_CLK	ALT7

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
UART3	ipp_uart_rts_b	Module: UARTV2, Protocol: DCE_MUX_FULL_ST, Port: RTS	USBH1_DIR	ALT7
	ipp_uart_rxd_mux	Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	EIM_D25	ALT3
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	EIM_D26	ALT3
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	AUD3_BB_RXD	ALT1
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	AUD3_BB_FS	ALT1
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	UART3_RXD	ALT1
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	UART3_TXD	ALT1
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	USBH1_STP	ALT5
UART3	ipp_uart_rxd_mux	Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	USBH1_NXT	ALT5
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: RXD_MUX	DISP2_DAT0	ALT5
		Module: UARTV2, Protocol: DCE_MUX_BASIC, Port: TXD_MUX	DISP2_DAT1	ALT5
USBOH3	ipp_ind_uh3_clk	Module: USBOH3, Protocol: ULPI, Port: USBH3_CLK	NANDF_RB3	ALT6
			DISP2_DAT0	ALT3
	ipp_ind_uh3_data_0	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA0	NANDF_D7	ALT5
			DISP2_DAT8	ALT3
	ipp_ind_uh3_data_1	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA1	NANDF_D6	ALT5

Table 4-3. i.MX51 Daisy Chain Settings (continued)

Instance	IN Pin	Module, Protocol, Port	Pad	Mode
USBOH3	ipp_ind_uh3_data_1	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA1	DISP2_DAT9	ALT3
	ipp_ind_uh3_data_2	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA2	NANDF_D5	ALT5
			DISP2_DAT10	ALT3
	ipp_ind_uh3_data_3	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA3	NANDF_D4	ALT5
			DISP2_DAT11	ALT3
	ipp_ind_uh3_data_4	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA4	NANDF_D3	ALT5
			DISP2_DAT12	ALT3
	ipp_ind_uh3_data_5	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA5	NANDF_D2	ALT5
USBOH3	ipp_ind_uh3_data_5	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA5	DISP2_DAT13	ALT3
	ipp_ind_uh3_data_6	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA6	NANDF_D1	ALT5
			DISP2_DAT14	ALT3
	ipp_ind_uh3_data_7	Module: USBOH3, Protocol: ULPI, Port: USBH3_DATA7	NANDF_D0	ALT5
			DISP2_DAT15	ALT3
	ipp_ind_uh3_dir	Module: USBOH3, Protocol: ULPI, Port: USBH3_DIR	NANDF_CS5	ALT7
			DISP2_DAT1	ALT3
	ipp_ind_uh3_nxt	Module: USBOH3, Protocol: ULPI, Port: USBH3_NXT	NANDF_RB2	ALT6
USBOH3	ipp_ind_uh3_nxt	Module: USBOH3, Protocol: ULPI, Port: USBH3_NXT	DISP2_DAT7	ALT3
	ipp_ind_uh3_stp	Module: USBOH3, Protocol: ULPI, Port: USBH3_STP	NANDF_CS4	ALT7
			DISP2_DAT6	ALT3

Chapter 5
External Memories

5.1 Overview

The M4IF is the module that controls all external memory accesses (read/write/erase/program) from all the masters in the system to different external memories. All accesses are arbitrated by the Multi Master Multi Memory Interface (M4IF) module and controlled by the respective memory controller. The high level block diagram is presented in Figure 5-1.

The Masters Interface to the EMI port is a full AXI interface with separated input bus for read and write access, so master that handle separated buses for read and write access can access the EMI. The data width can be 32 or 64 bits. All ports support 32 or 64-bit IF.

5.2 External Memory Interface

The M4IF provides the ability to connect to a wide variety of memory devices. This chapter contains the technical information about the operation and configuration of the M4IF modules in the chip to allow the designer to quickly integrate external memory devices into new and existing designs. Pin sharing is done between the data bus of the WEIM and the Nand Flash Controller in order to reduce the total number of pins needed for the M4IF.

The EMI is an External Memory Interface and arbitration between multi AXI masters to multi memory controllers, divided into three major channels, fast memories (Mobile DDR SDRAM) channel, slow memories (NOR-FLASH/PSRAM/NAND-FLASH etc.) channel, Internal Memory (RAM, ROM) channel and Internal GMEM memory.

To increase the bandwidth performance, the EMI separates the buffering and the arbitration between accesses to fast channel slow channel and Internal Memory channels, so parallel accesses can occur. By separating the three channels, slow accesses do not interfere with fast accesses.

The M4IF contains the arbitration interface and different external memory controllers to support the following memory devices:

- M4IF—Multi Master Multi Memory Interface
- ESDRAMC—Enhanced Mobile LPDDR SDRAM memory controller
- NFC—NAND Flash memory controller
- WEIM—SRAM/PSRAM/NOR Flash memory controller

5.2.1 EMI i.MX51 Masters

Table 5-1 provides details on EMI masters and the EMI port associated with each master.

Table 5-1. AMXI Bus Masters

Module	Direct Bus	Master Port	Buffered	Boundary Crossing
ARM Cortex-A8	AXI	3	Y	4K
SDMA (burst)	AHB via AHBS2AXI	4	N	1K
SDMA (non-burst)	AHB via AHBMAX	2	N	1K
IPUEX	AXI	0	Y	4K
VPU	AXI	1	Y	4K
GPU3D	AXI	5	Y	4K
RTIC	AHB via AHBMAX	2	N	1K
SCC	AHB via AHBMAX	2	N	1K
USBOH3	AXI	7	N	4K
GPU2D (OpenVG)	AXI	6	Y	4K
FEC	AHB via AHBMAX	2	N	1K
SAHARA	AHB via AHBMAX	2	N	1K
eSDHC 1	AHB via AHBMAX	2	N	1K
eSDHC 2	AHB via AHBMAX	2	N	1K
eSDHC 4	AHB via AHBMAX	2	N	1K
eSDHC 3 - CE-ATA	AHB via AHBMAX	2	N	1K
DAP	AHB via AHBMAX	2	N	1K

5.2.2 Features

Each of the EMI memory controller block guides specify detailed information about the supported features, programming model. However, in i.MX51 some of those feature are disabled or not supported.

The M4IF in the i.MX51 includes these distinctive features:

- Multi Master Multi Memory Interface (M4IF)
  - — Supports multiple accesses from 8 masters through different input ports interfaces. Each port can support either of the following two data width options:
    - – ×32 AXI port.
    - – ×64 AXI port.
  - — Supports different clock domain for each AXI port master
  - — Configurable memory “snooping”
  - — Configurable memory watermark protection per CS

- Enhanced arbitration scheme for fast channel, consider page hit/miss, last access details (read/write) and fixed priority configuration

- Enhanced SDRAM Controller (ESDRAMC) or LPDDR Controller (LPDDRC)
  - Up to 2 chip selects support up to 256 MBytes each
  - ×64 AXI port
  - Supports ×16/×32 LPDDR/Non-mobile DDR1 SDRAM at clock frequency up to 200 MHz (DDR400)
  - Supports ×16/×32 DDR2 memories up to 200 MHz (400 MHz data rate)
  - Supports latency hiding logic
- NANDFlash Controller (NFC)
  - ×8/×16 NAND interface
  - Up to 8 chip selects support up to 8 Gbit in 1/2K page mode, 64 Gbit in 2K page mode and 256 Gbit in 4K page mode each.
  - 4.5K RAM Internal Buffer
  - MLC and SLC memory support.
  - Configurable operation mode—symmetric and asymmetric
  - Configurable page mode, 1/2K, 2K, or 4K
  - ECC support up to 8 bits
  - Supports up to 8 mutually exclusive, yet interleaved NAND devices.
  - Automatic Common Nand Operations
- Wireless External Interface Memory Controller (WEIM)
  - Up to 6 chip selects.
  - Supports ×32/×16 PSRAM (up to 133 MHz).
  - Supports ×32/×16 muxed mode PSRAM / NOR (up to 133 MHz).
  - Supports ×32/×16 NOR (up to 133 MHz).
- Supports DVFS, voltage and frequency change.
- Supports watermark configuration.
- Supports automatic shut-down (power saving features)
- Enhanced debug capabilities (trace mode).

5.3 M4IF Setup

The following section describes i.MX51 specific requirements in order to use the M4IF module.

5.3.1 Clock Domains

The EMI contains the following clock domains:

- EMI IPS clock
- ESDCTL main clock (up to 200 MHz)

- Slow arbitration clock (up to 133 MHz)
- Internal memory arbitration 1 and 2 clocks (up to 166 MHz)
- 8 × masters clocks, can be asynchronous to the EMI arbitration clocks (66 MHz–166 MHz)
- NFC main clock—should be integer divided from slow arbitration clock (up to 50 MHz)
- SDCLK—SDR/DDR clock to SDR/DDR SDRAM device
- BCLK—NOR Flash/PSRAM clock WEIM in synchronous mode

5.3.2 Boot Scenarios

The M4IF memory controllers allow booting from the following memories: NOR Flash devices through the WEIM and NAND Flash devices through the NAND Flash Controller. Special signals coming from the SOC define the different booting options to those memories like the memory data width, memory page size, and other specific parameters for the initial access of the boot. For more details refer to the memory controllers detailed chapters and boot chapter.

5.3.3 Watermark Ports

Watermark regions are supported in the M4IF while the configuration is handled by the CSU module. The external memory spaces have trust zone area's that only trust zone accesses are able to reach. Non-trust zone accesses are blocked. The CSU module has the registers that define those regions. It sends the indication of the region and whether the specific access is trust zone or not to the M4IF. The EMI is also capable of handling the special watermark zone for the BP domain, but this is not used for the i.MX51.

The watermark indication is dynamic and can change for each access.

For details on the watermark configuration ,refer to the CSU chapter.

5.3.4 Drive Strength Settings

The IOMUXC registers that controls the ESDRAMC relevant BGA contact drive strength are specified in Table 5-2.

Table 5-2. Drive Strength Settings

Functional Group	Signal Group		Drive Strength Mode			
	BGA Contact name	Registers settings	MAX	HIGH	MEDIUM	LOW
DATA	DRAM_D[7:0]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DRAM_B0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_D[15:8]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DRAM_B1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_D[23:16]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DRAM_B2			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_D[31:24]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DRAM_B4			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
DATA MASK	DRAM_DQM[0]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_DQM[1]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_DQM[2]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_DQM[3]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
DATA QUALIFIERS	DRAM_SDQS[0]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_SDQS[1]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_SDQS[2]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_SDQS[3]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
ADDRESS	DRAM_A[7:0]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DDR_A0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	DRAM_A[14:8], SDBA[2:0]	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_GRP_DDR_A1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000

Table 5-2. Drive Strength Settings (continued)

Functional Group	Signal Group		Drive Strength Mode			
	BGA Contact name	Registers settings	MAX	HIGH	MEDIUM	LOW
CLOCKS	SDCLK	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
CONTROLS	RAS	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	CAS	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	SDWE	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	SDCKE0	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	SDCKE1	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	CS0	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000
	CS1	Address	0xIOMUXC_BASE+IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1			
		Value	0x0000_0006	0x0000_0004	0x0000_0002	0x0000_0000

5.3.5 M4IF I/O MUX

The M4IF provides the ability to mux some of its signals in the SOC to allow pin sharing between the different memory controllers. Muxing is possible on the slow channel only because of timing limitations.

Only shared i.MX51 PADS signals/buses are routed through the I/O MUX toward the external devices. Signals (mainly controls) that have dedicated PADS are directly routed from the memory controllers to the external devices. Table 5-3 summarizes the EMI PADS definition in i.MX51.

Table 5-3. EMI IOMUX in i.MX51

EMI Port	Contact	Mode
DRAM_A[0]	DRAM_A0	No Muxing (ALT0)
DRAM_A[10]	DRAM_A10	No Muxing (ALT0)
DRAM_A[11]	DRAM_A11	No Muxing (ALT0)
DRAM_A[12]	DRAM_A12	No Muxing (ALT0)
DRAM_A[13]	DRAM_A13	No Muxing (ALT0)
DRAM_A[14]	DRAM_A14	No Muxing (ALT0)

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
DRAM_A[1]	DRAM_A1	No Muxing (ALT0)
DRAM_A[2]	DRAM_A2	No Muxing (ALT0)
DRAM_A[3]	DRAM_A3	No Muxing (ALT0)
DRAM_A[4]	DRAM_A4	No Muxing (ALT0)
DRAM_A[5]	DRAM_A5	No Muxing (ALT0)
DRAM_A[6]	DRAM_A6	No Muxing (ALT0)
DRAM_A[7]	DRAM_A7	No Muxing (ALT0)
DRAM_A[8]	DRAM_A8	No Muxing (ALT0)
DRAM_A[9]	DRAM_A9	No Muxing (ALT0)
DRAM_CAS	DRAM_CAS	No Muxing (ALT0)
DRAM_CS0	DRAM_CS0	No Muxing (ALT0)
DRAM_CS1	DRAM_CS1	ALT0
DRAM_DQM[0]	DRAM_DQM0	No Muxing (ALT0)
DRAM_DQM[1]	DRAM_DQM1	No Muxing (ALT0)
DRAM_DQM[2]	DRAM_DQM2	No Muxing (ALT0)
DRAM_DQM[3]	DRAM_DQM3	No Muxing (ALT0)
DRAM_D[0]	DRAM_D0	No Muxing (ALT0)
DRAM_D[10]	DRAM_D10	No Muxing (ALT0)
DRAM_D[11]	DRAM_D11	No Muxing (ALT0)
DRAM_D[12]	DRAM_D12	No Muxing (ALT0)
DRAM_D[13]	DRAM_D13	No Muxing (ALT0)
DRAM_D[14]	DRAM_D14	No Muxing (ALT0)
DRAM_D[15]	DRAM_D15	No Muxing (ALT0)
DRAM_D[16]	DRAM_D16	No Muxing (ALT0)
DRAM_D[17]	DRAM_D17	No Muxing (ALT0)
DRAM_D[18]	DRAM_D18	No Muxing (ALT0)
DRAM_D[19]	DRAM_D19	No Muxing (ALT0)
DRAM_D[1]	DRAM_D1	No Muxing (ALT0)
DRAM_D[20]	DRAM_D20	No Muxing (ALT0)
DRAM_D[21]	DRAM_D21	No Muxing (ALT0)
DRAM_D[22]	DRAM_D22	No Muxing (ALT0)
DRAM_D[23]	DRAM_D23	No Muxing (ALT0)
DRAM_D[24]	DRAM_D24	No Muxing (ALT0)

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
DRAM_D[25]	DRAM_D25	No Muxing (ALT0)
DRAM_D[26]	DRAM_D26	No Muxing (ALT0)
DRAM_D[27]	DRAM_D27	No Muxing (ALT0)
DRAM_D[28]	DRAM_D28	No Muxing (ALT0)
DRAM_D[29]	DRAM_D29	No Muxing (ALT0)
DRAM_D[2]	DRAM_D2	No Muxing (ALT0)
DRAM_D[30]	DRAM_D30	No Muxing (ALT0)
DRAM_D[31]	DRAM_D31	No Muxing (ALT0)
DRAM_D[3]	DRAM_D3	No Muxing (ALT0)
DRAM_D[4]	DRAM_D4	No Muxing (ALT0)
DRAM_D[5]	DRAM_D5	No Muxing (ALT0)
DRAM_D[6]	DRAM_D6	No Muxing (ALT0)
DRAM_D[7]	DRAM_D7	No Muxing (ALT0)
DRAM_D[8]	DRAM_D8	No Muxing (ALT0)
DRAM_D[9]	DRAM_D9	No Muxing (ALT0)
DRAM_ODT0	EIM_SDODT0	No Muxing (ALT0)
DRAM_ODT1	EIM_SDODT1	No Muxing (ALT0)
DRAM_RAS	DRAM_RAS	No Muxing (ALT0)
DRAM_SDBA[0]	EIM_SDBA0	No Muxing (ALT0)
DRAM_SDBA[1]	EIM_SDBA1	No Muxing (ALT0)
DRAM_SDBA[2]	EIM_SDBA2	No Muxing (ALT0)
DRAM_SDCKE[0]	DRAM_SDCKE0	No Muxing (ALT0)
DRAM_SDCKE[1]	DRAM_SDCKE1	No Muxing (ALT0)
DRAM_SDCLK	DRAM_SDCLK	No Muxing (ALT0)
DRAM_SDCLK_B	DRAM_SDCLK_B	No Muxing (ALT0)
DRAM_SDQS[0]	DRAM_SDQS0	No Muxing (ALT0)
DRAM_SDQS[1]	DRAM_SDQS1	No Muxing (ALT0)
DRAM_SDQS[2]	DRAM_SDQS2	No Muxing (ALT0)
DRAM_SDQS[3]	DRAM_SDQS3	No Muxing (ALT0)
DRAM_SDQS_B[0]	DRAM_SDQS0_B	No Muxing (ALT0)
DRAM_SDQS_B[1]	DRAM_SDQS1_B	No Muxing (ALT0)
DRAM_SDQS_B[2]	DRAM_SDQS2_B	No Muxing (ALT0)
DRAM_SDQS_B[3]	DRAM_SDQS3_B	No Muxing (ALT0)

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
DRAM_SDWE	DRAM_SDWE	No Muxing (ALT0)
DSTROBE	KEY_ROW0	ALT1
EIM_A[16]	EIM_A16	ALT0
EIM_A[17]	EIM_A17	ALT0
EIM_A[18]	EIM_A18	ALT0
EIM_A[19]	EIM_A19	ALT0
EIM_A[20]	EIM_A20	ALT0
EIM_A[21]	EIM_A21	ALT0
EIM_A[22]	EIM_A22	ALT0
EIM_A[23]	EIM_A23	ALT0
EIM_A[24]	EIM_A24	ALT0
EIM_A[25]	EIM_A25	ALT0
EIM_A[26]	EIM_A26	ALT0
EIM_A[27]	EIM_A27	ALT0
EIM_BCLK	EIM_BCLK	No Muxing (ALT0)
EIM_CRE	EIM_CRE	ALT0
EIM_CS0	EIM_CS0	ALT0
EIM_CS1	EIM_CS1	ALT0
EIM_CS2	EIM_CS2	ALT0
EIM_CS3	EIM_CS3	ALT0
EIM_CS4	EIM_CS4	ALT0
EIM_CS5	EIM_CS5	ALT0
EIM_DA[0]	EIM_DA0	ALT0
EIM_DA[10]	EIM_DA10	ALT0
EIM_DA[11]	EIM_DA11	ALT0
EIM_DA[12]	EIM_DA12	ALT0
EIM_DA[13]	EIM_DA13	ALT0
EIM_DA[14]	EIM_DA14	ALT0
EIM_DA[15]	EIM_DA15	ALT0
EIM_DA[1]	EIM_DA1	ALT0
EIM_DA[2]	EIM_DA2	ALT0
EIM_DA[3]	EIM_DA3	ALT0
EIM_DA[4]	EIM_DA4	ALT0

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
EIM_DA[5]	EIM_DA5	ALT0
EIM_DA[6]	EIM_DA6	ALT0
EIM_DA[7]	EIM_DA7	ALT0
EIM_DA[8]	EIM_DA8	ALT0
EIM_DA[9]	EIM_DA9	ALT0
EIM_EB[0]	EIM_EB0	ALT0
EIM_EB[1]	EIM_EB1	ALT0
EIM_EB[2]	EIM_EB2	ALT0
EIM_EB[3]	EIM_EB3	ALT0
EIM_LBA	EIM_LBA	ALT0
EIM_NFC_D[0]	NANDF_D0	ALT0
EIM_NFC_D[10]	NANDF_D10	ALT0
EIM_NFC_D[11]	NANDF_D11	ALT0
EIM_NFC_D[12]	NANDF_D12	ALT0
EIM_NFC_D[13]	NANDF_D13	ALT0
EIM_NFC_D[14]	NANDF_D14	ALT0
EIM_NFC_D[15]	NANDF_D15	ALT0
EIM_NFC_D[1]	NANDF_D1	ALT0
EIM_NFC_D[2]	NANDF_D2	ALT0
EIM_NFC_D[3]	NANDF_D3	ALT0
EIM_NFC_D[4]	NANDF_D4	ALT0
EIM_NFC_D[5]	NANDF_D5	ALT0
EIM_NFC_D[6]	NANDF_D6	ALT0
EIM_NFC_D[7]	NANDF_D7	ALT0
EIM_NFC_D[8]	NANDF_D8	ALT0
EIM_NFC_D[9]	NANDF_D9	ALT0
EIM_OE	EIM_OE	ALT0
EIM_RW	EIM_RW	No Muxing (ALT0)
EIM_WAIT	EIM_WAIT	No Muxing (ALT0)
EMI_DEBUG[0]	USBH1_CLK	ALT6
EMI_DEBUG[10]	USBH1_DATA6	ALT6
EMI_DEBUG[11]	USBH1_DATA7	ALT6
EMI_DEBUG[12]	UART1_RXD	ALT6

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
EMI_DEBUG[13]	UART1_TXD	ALT6
EMI_DEBUG[14]	UART1_RTS	ALT6
EMI_DEBUG[15]	UART1_CTS	ALT6
EMI_DEBUG[16]	UART2_RXD	ALT6
EMI_DEBUG[17]	UART2_TXD	ALT6
EMI_DEBUG[18]	UART3_RXD	ALT6
EMI_DEBUG[19]	UART3_TXD	ALT6
EMI_DEBUG[1]	USBH1_DIR	ALT6
EMI_DEBUG[20]	KEY_ROW0	ALT6
EMI_DEBUG[21]	KEY_ROW1	ALT6
EMI_DEBUG[22]	KEY_ROW2	ALT6
EMI_DEBUG[23]	KEY_ROW3	ALT6
EMI_DEBUG[24]	DI1_PIN3	ALT6
EMI_DEBUG[25]	DI1_PIN2	ALT6
EMI_DEBUG[26]	DI_GP1	ALT6
EMI_DEBUG[27]	DI_GP2	ALT6
EMI_DEBUG[28]	DI_GP3	ALT6
EMI_DEBUG[29]	DI2_PIN4	ALT6
EMI_DEBUG[2]	USBH1_STP	ALT6
EMI_DEBUG[30]	DI2_PIN2	ALT6
EMI_DEBUG[31]	DI2_PIN3	ALT6
EMI_DEBUG[32]	DI_GP4	ALT6
EMI_DEBUG[33]	DISP2_DAT0	ALT6
EMI_DEBUG[34]	DISP2_DAT1	ALT6
EMI_DEBUG[35]	DISP2_DAT6	ALT6
EMI_DEBUG[36]	DISP2_DAT7	ALT6
EMI_DEBUG[37]	DISP2_DAT8	ALT6
EMI_DEBUG[38]	DISP2_DAT9	ALT6
EMI_DEBUG[39]	DISP2_DAT10	ALT6
EMI_DEBUG[3]	USBH1_NXT	ALT6
EMI_DEBUG[40]	DISP2_DAT11	ALT6
EMI_DEBUG[41]	DISP2_DAT12	ALT6
EMI_DEBUG[42]	DISP2_DAT13	ALT6

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
EMI_DEBUG[43]	DISP2_DAT14	ALT6
EMI_DEBUG[44]	DISP2_DAT15	ALT6
EMI_DEBUG[45]	CSI2_D13	ALT6
EMI_DEBUG[46]	CSI2_D18	ALT6
EMI_DEBUG[47]	CSI2_D19	ALT6
EMI_DEBUG[48]	CSI2_VSYNC	ALT6
EMI_DEBUG[49]	CSI2_HSYNC	ALT6
EMI_DEBUG[4]	USBH1_DATA0	ALT6
EMI_DEBUG[50]	CSI2_PIXCLK	ALT6
EMI_DEBUG[5]	USBH1_DATA1	ALT6
EMI_DEBUG[6]	USBH1_DATA2	ALT6
EMI_DEBUG[7]	USBH1_DATA3	ALT6
EMI_DEBUG[8]	USBH1_DATA4	ALT6
EMI_DEBUG[9]	USBH1_DATA5	ALT6
NANDF_ALE	NANDF_ALE	ALT0
NANDF_CLE	NANDF_CLE	ALT0
NANDF_CS0	NANDF_CS0	ALT0
NANDF_CS1	NANDF_CS1	ALT0
NANDF_CS2	NANDF_CS2	ALT0
NANDF_CS3	NANDF_CS3	ALT0
NANDF_CS4	NANDF_CS4	ALT0
NANDF_CS5	NANDF_CS5	ALT0
NANDF_CS6	NANDF_CS6	ALT0
NANDF_CS7	NANDF_CS7	ALT0
NANDF_RB0	NANDF_RB0	ALT0
NANDF_RB1	NANDF_RB1	ALT0
NANDF_RB2	NANDF_RB2	ALT0
NANDF_RB3	NANDF_RB3	ALT0
NANDF_RE_B	NANDF_RE_B	ALT0
NANDF_WE_B	NANDF_WE_B	ALT0
NANDF_WP_B	NANDF_WP_B	ALT0
RDY	GPIO1_4	ALT3
	NANDF_RDY_INT	ALT0

Table 5-3. EMI IOMUX in i.MX51 (continued)

EMI Port	Contact	Mode
WEIM_DTACK_B	EIM_DTACK	ALT0
WEIM_D[16]	EIM_D16	ALT0
WEIM_D[17]	EIM_D17	ALT0
WEIM_D[18]	EIM_D18	ALT0
WEIM_D[19]	EIM_D19	ALT0
WEIM_D[20]	EIM_D20	ALT0
WEIM_D[21]	EIM_D21	ALT0
WEIM_D[22]	EIM_D22	ALT0
WEIM_D[23]	EIM_D23	ALT0
WEIM_D[24]	EIM_D24	ALT0
WEIM_D[25]	EIM_D25	ALT0
WEIM_D[26]	EIM_D26	ALT0
WEIM_D[27]	EIM_D27	ALT0
WEIM_D[28]	EIM_D28	ALT0
WEIM_D[29]	EIM_D29	ALT0
WEIM_D[30]	EIM_D30	ALT0
WEIM_D[31]	EIM_D31	ALT0

5.4 External Memory Controllers Preset Operation

This section discusses the preset operations of the following:

- NAND Flash Controller
- WEIM
- ESDRAMC
- M4IF

5.4.1 NAND Flash Controller (NFC) Preset Operation

The NFC has several inputs that defines the initial operation of this module. This information is the page size (0.5K, 2K, 4K), data size (8/16 bit), and NFC clock configuration. Those parameters are defined by fuse programing.

For initialization examples, please refer to the Nand Flash Controller specific chapter.

5.4.2 WEIM Preset Operation

The initial operation of the WEIM is defined by the memory data width, muxed/non-muxed memory, address unshift, and merged CS0 and CS1 memory space. These inputs define how the WEIM operates during direct boot to NOR Flash devices.

For an initialization example, please refer to the WEIM Controller specific chapter.

5.4.3 ESDRAMC Preset Operation

The ESDRAMC functionality depends on the register setup that defines the derive strength of the pads related to it (see Table 5-2). For the detailed needs of the DDR delay line and options for frequency changes, refer to the specific ESDCTL chapter.

Refer to the ESDCTL chapter for an example for ESDCTL initialization.

5.4.4 M4IF Preset Operation

Security levels, locking indications, and endianness define the way the M4IF works. Refer to the M4IF specific chapter for a detailed description.

Chapter 6
Fuse Map

6.1 Overview

This chapter contains the fuse map for the i.MX51. There are four fuse banks. Many fuse rows may be be locked by setting a single fuse.

6.1.1 Fuse Locks

Table 6-1 is a summary of the fuse locks used in the i.MX51.

Table 6-1. Fuse Lock Summary

Fuse Name	Fuse Bank	Rows Affected
IMEI_LOCK	0	0860–087C
BOOT_LOCK	0	0804, 080C–0818, 0840–0844, 0854
MAC_ADDR_LOCK	1	0824–0838
SRK_LOCK	1	0C04
SJC_RESP_LOCK	1	0C08–0C20
TRIM_LOCK	1	0C24–0C7C
SRK_LOCK88	3	1404–142C
SRK_LOCK160	3	1430–147C

6.1.2 Fuse Map

Table 6-2, beginning on page 6-2, is a complete listing of all user-accessible fuses.

Table 6-2. i.MX51 Fuse Map

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
0	0800	FBWP	FBOP	FBRP	Reserved	FBESP	Reserved	IMEI_ LOCK	BOOT_ LOCK	0001 0000	—	—
0	0804	OSC_ FREQ_ SEL[1]	JTAG_ SMODE [1:0]		OSC_ FREQ_ SEL[0]	JTAG_ HEO	KTE	SEC_ JTAG_RE	JTAG_BP	0000 0000	BOOT_ LOCK	note[2] note [3]
0	0808	Freescale Internal Use								xxxx xxxx	BOOT_ LOCK	—
0	080C	BT_ SRC[1:0]		BT_ MLC_SEL	BT_WEIM_ MUXED[1:0]		BT_ SPARE_ SIZE	BT_ DPLUS_ BYPASS	BT_USB_ SRC	0000 0000	BOOT_ LOCK	note[2]
0	0810	BT_ UNPROGR AMMED	BT_PAGE_SIZE[1:0]		BT_ EEPROM _CFG	GPIO_ BT_SEL	HAB_ TYPE[2]	HAB_ TYPE[1]	BT_ EEPROM _CFG	0000 0001	BOOT_ LOCK	
0	0814	BT_UART_SRC[1:0]		BT_MEM_TYPE[1:0]		BT_BUS_ WIDTH	BT_MEM_CTL[1:0]		DIR_BT_ DIS	0000 0000	BOOT_ LOCK	
0	0818	HAB_CUS[7:0]								0000 0000	BOOT_ LOCK	—
0	081C	Freescale Internal Use								xxxx 0000	—	—
0	0820	Freescale Internal Use								xxxx xxxx	—	—
0	0824	Freescale Internal Use								xxxx xxxx	—	—
0	0828	Freescale Internal Use								xxxx xxxx	—	—
0	082C	Freescale Internal Use								xxxx xxxx	—	—
0	0830	Freescale Internal Use								xxxx xxxx	—	—
0	0834	Freescale Internal Use								xxxx xxxx	—	—
0	0838	Freescale Internal Use								xxxx xxxx	—	—
0	083C	Freescale Internal Use								xxxx xxxx	—	—
0	0840	SRTC_MCOUNT[2:0]			CMD_ DEFAULT	BT_LPB[1:0]		SRTC_SECMODE[1:0]		0000 0000	BOOT_ LOCK	note[2]

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
0	0844	BT_LPB_ FREQ[2:0]			CSU_ FA_OUT[1:0]		CSU_AM_DIS[1:0]		CSU_ FA_ COUNT	0000 0000	BOOT_ LOCK	note[2]
0	0848	AP_BI_VER[15:8]								0000 0000	—	—
0	084C	AP_BI_VER[7:0]								0000 0000	—	—
0	0850	Freescale Internal Use								xxxx xxxx	—	—
0	0854	Freescale Internal Use	SJC_DIS ABLE	Freescale Internal Use						1000 0000	BOOT_ LOCK	—
0	0858	Freescale Internal Use								xxxx xxxx	—	—
0	085C	Freescale Internal Use								xxxx xxxx	—	—
0	0860	IMEI[63:56]								0000 0000	IMEI_ LOCK	—
0	0864	IMEI[55:48]								0000 0000	IMEI_ LOCK	—
0	0868	IMEI[47:40]								0000 0000	IMEI_ LOCK	—
0	086C	IMEI[39:32]								0000 0000	IMEI_ LOCK	—
0	0870	IMEI[31:24]								0000 0000	IMEI_ LOCK	—
0	0874	IMEI[23:16]								0000 0000	IMEI_ LOCK	—
0	0878	IMEI[15:8]								0000 0000	IMEI_ LOCK	—
0	087C	IMEI[7:0]								0000 0000	IMEI_ LOCK	—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
1	0C00	FBWP	FBOP	FBRP	MAC_ ADDR_ LOCK	FBESP	SRK_ LOCK	SJC_ RESP_ LOCK	SCC_ LOCK	0000 0001	—	—
1	0C04	SRK_HASH[255:248]								0000 0000	SRK_ LOCK	note[4]
1	0C08	SJC_RESP[55:48]								0000 0000	SJC_ RESP	note[5]
1	0C0C	SJC_RESP[47:40]								0000 0000	SJC_ RESP	
1	0C10	SJC_RESP[39:32]								0000 0000	SJC_ RESP	
1	0C14	SJC_RESP[31:24]								0000 0000	SJC_ RESP	
1	0C18	SJC_RESP[23:16]								0000 0000	SJC_ RESP	
1	0C1C	SJC_RESP[15:8]								0000 0000	SJC_ RESP	
1	0C20	SJC_RESP[7:0]								0000 0000	SJC_ RESP	
1	0C24	MAC_ADDR[47:40]								0000 0000	MAC_ ADDR_ LOCK	—
1	0C28	MAC_ADDR[39:32]								0000 0000	MAC_ ADDR_ LOCK	—
1	0C2C	MAC_ADDR[31:24]								0000 0000	MAC_ ADDR_ LOCK	—
1	0C30	MAC_ADDR[23:16]								0000 0000	MAC_ ADDR_ LOCK	—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
1	0C34	MAC_ADDR[15:8]								0000 0000	MAC_ADDR_LOCK	—
1	0C38	MAC_ADDR[7:0]								0000 0000	MAC_ADDR_LOCK	—
1	0C3C	Freescale Internal		TVDAC_GAIN1[5:0]						0000 0000	TRIM_LOCK	—
1	0C40	Freescale Internal		TVDAC_GAIN2[5:0]						0000 0000	TRIM_LOCK	—
1	0C44	Freescale Internal		TVDAC_GAIN3[5:0]						0000 0000	TRIM_LOCK	—
1	0C48	PTC_VER[2:0]			GDPTCV_VALID	GDPTCV[3:0]				0000 0000	TRIM_LOCK	—
1	0C4C	MMU_EN	Freescale Internal		LDPTCV_VALID	LDPTCV[3:0]				0000 0000	TRIM_LOCK	—
1	0C50	DVFS_DELAY_ADJUST[7:0]								0000 0000	TRIM_LOCK	—
1	0C54	Freescale Internal		l1d_tch[2:0]			l1d_tcs[2:0]			0000 0000	TRIM_LOCK	—
1	0C58	tlb_tc[1:0]		l1i_tch[2:0]			l1i_tcs[2:0]			0000 0000	TRIM_LOCK	—
1	0C5C	Freescale Internal Use								xxxx xxxx	TRIM_LOCK	—
1	0C60	Freescale Internal Use								xxxx xxxx	TRIM_LOCK	—
1	0C64	Freescale Internal Use								xxxx xxxx	TRIM_LOCK	—
1	0C68	Freescale Internal Use								xxxx xxxx	TRIM_LOCK	—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
1	0C6C	Freescale Internal Use								xxxx xxxx	TRIM_ LOCK	—
1	0C70	Freescale Internal Use								xxxx xxxx	TRIM_ LOCK	—
1	0C74	Freescale Internal Use								xxxx xxxx	TRIM_ LOCK	—
1	0C78	Freescale Internal Use								xxxx xxxx	TRIM_ LOCK	—
1	0C7C	Freescale Internal Use								xxxx xxxx	TRIM_ LOCK	—
2	1000	Freescale Internal Use								xxxx xxxx		—
2	1004	Freescale Internal Use								xxxx xxxx		—
2	1008	Freescale Internal Use								xxxx xxxx		—
2	100C	Freescale Internal Use								xxxx xxxx		—
2	1010	Freescale Internal Use								xxxx xxxx		—
2	1014	Freescale Internal Use								xxxx xxxx		—
2	1018	Freescale Internal Use								xxxx xxxx		—
2	101C	Freescale Internal Use								xxxx xxxx		—
2	1020	Freescale Internal Use								xxxx xxxx		—
2	1024	Freescale Internal Use								xxxx xxxx		—
2	1028	Freescale Internal Use								xxxx xxxx		—
2	102C	Freescale Internal Use								xxxx xxxx		—
2	1030	Freescale Internal Use								xxxx xxxx		—
2	1034	Freescale Internal Use								xxxx xxxx		—
2	1038	Freescale Internal Use								xxxx xxxx		—
2	103C	Freescale Internal Use								xxxx xxxx		—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
2	1040	Freescale Internal Use								xxxx xxxx		—
2	1044	Freescale Internal Use								xxxx xxxx		—
2	1048	Freescale Internal Use								xxxx xxxx		—
2	104C	Freescale Internal Use								xxxx xxxx		—
2	1050	Freescale Internal Use								xxxx xxxx		—
2	1054	Freescale Internal Use								xxxx xxxx		—
2	1058	Freescale Internal Use								xxxx xxxx		—
2	105C	Freescale Internal Use								xxxx xxxx		—
2	1060	Freescale Internal Use								xxxx xxxx		—
2	1064	Freescale Internal Use								xxxx xxxx		—
2	1068	Freescale Internal Use								xxxx xxxx		—
2	106C	Freescale Internal Use								xxxx xxxx		—
2	1070	Freescale Internal Use								xxxx xxxx		—
2	1074	Freescale Internal Use								xxxx xxxx		—
2	1078	Freescale Internal Use								xxxx xxxx		—
2	107C	Freescale Internal Use								xxxx xxxx		—
3	1400	FBWP	FBOP	FBRP	TRIM_ LOCK	FBESP	Reserved for customer	SRK_ LOCK88	SRK_LOC K160	0000 0000	—	—
3	1404	SRK_HASH[247:240]								0000 0000	SRK_ LOCK88	—
3	1408	SRK_HASH[239:232]								0000 0000	SRK_ LOCK88	—
3	140C	SRK_HASH[231:224]								0000 0000	SRK_ LOCK88	—
3	1410	SRK_HASH[223:216]								0000 0000	SRK_ LOCK88	—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
3	1414	SRK_HASH[215:208]								0000 0000	SRK_ LOCK88	—
3	1418	SRK_HASH[207:200]								0000 0000	SRK_ LOCK88	—
3	141C	SRK_HASH[199:192]								0000 0000	SRK_ LOCK88	—
3	1420	SRK_HASH[191:184]								0000 0000	SRK_ LOCK88	—
3	1424	SRK_HASH[183:176]								0000 0000	SRK_ LOCK88	—
3	1428	SRK_HASH[175:168]								0000 0000	SRK_ LOCK88	—
3	142C	SRK_HASH[167:160]								0000 0000	SRK_ LOCK88	—
3	1430	SRK_HASH[159:152]								0000 0000	SRK_ LOCK160	—
3	1434	SRK_HASH[151:144]								0000 0000	SRK_ LOCK160	—
3	1438	SRK_HASH[143:136]								0000 0000	SRK_ LOCK160	—
3	143C	SRK_HASH[135:128]								0000 0000	SRK_ LOCK160	—
3	1440	SRK_HASH[127:120]								0000 0000	SRK_ LOCK160	—
3	1444	SRK_HASH[119:112]								0000 0000	SRK_ LOCK160	—
3	1448	SRK_HASH[111:104]								0000 0000	SRK_ LOCK160	—
3	144C	SRK_HASH[103:96]								0000 0000	SRK_ LOCK160	—

Table 6-2. i.MX51 Fuse Map (continued)

Fuse Bank	Address[1]	7	6	5	4	3	2	1	0	Burned Value	Locked by	Comment
3	1450	SRK_HASH[95:88]								0000 0000	SRK_ LOCK160	—
3	1454	SRK_HASH[87:80]								0000 0000	SRK_ LOCK160	—
3	1458	SRK_HASH[79:72]								0000 0000	SRK_ LOCK160	—
3	145C	SRK_HASH[71:64]								0000 0000	SRK_ LOCK160	—
3	1460	SRK_HASH[63:56]								0000 0000	SRK_ LOCK160	—
3	1464	SRK_HASH[55:48]								0000 0000	SRK_ LOCK160	—
3	1468	SRK_HASH[47:40]								0000 0000	SRK_ LOCK160	—
3	146C	SRK_HASH[39:32]								0000 0000	SRK_ LOCK160	—
3	1470	SRK_HASH[31:24]								0000 0000	SRK_ LOCK160	—
3	1474	SRK_HASH[23:16]								0000 0000	SRK_ LOCK160	—
3	1478	SRK_HASH[15:8]								0000 0000	SRK_ LOCK160	—
3	147C	SRK_HASH[7:0]								0000 0000	SRK_ LOCK160	—

1 Address offset from bank base address
2 Shaded areas in the table indicate fuses that have corresponding GPIO pins
3 Controls JTAG debug operating modes
4 MSB (Most Significant Byte) of 256-bit of AP SRK HASH
5 SJC_RESP fuses that have being locked for read, explicit sense, override and writting.

Chapter 7
Clock Controller Module (CCM)

The Clock Controller Module (CCM) controls the clocks for the i.MX51 modules. This module uses the available clock sources to generate the clock roots. Figure 7-1 shows the CCM block diagram.

7.1 Overview

The Clock Controller Module controls the following functions in the i.MX51:

- Uses the available clock sources to generate clock roots to various parts of the SoC.
- Programable bits are used to control frequencies of the clock roots.
- Control of the low-power mechanism.
- Provides control signals to LPCG for gating clocks.
- Provides handshake with SRC for controlling clocks during reset.
- Provides handshake with GPC for support of DVFS, DPTC, and power gating operations.

7.1.1 Features

The CCM includes the following features:

- Clock switching module to generate three source clocks using three PLLs.
- Root clocks generation—provides root clock to the i.MX51's modules based on three switchable source clocks.
- ARM core root clock generated from a dedicated switchable source clock.
- Includes separate dividers to control generation of core and bus root clocks (axi's, ahb, ipg).
- Includes separate dividers and clock sources selectors for each serial root clock.
- Option for external clock to bypass PLLs clocks.
- Selects which clocks are routed to the IOMUX signals CLKO and CLKO2 for observability.
- Controllable registers are accessible via IP bus.
- Manages the Low-Power Modes, namely RUN, WAIT, SCREEN REFRESH and STOP. The gating of the peripheral clocks is programmable in RUN and WAIT modes.
- Manages frequency scaling procedure for ARM core clock by shifting between PLL sources, without loss of clocks.
- Manages frequency scaling procedure for peripheral clock roots by programable divider. The division occurs on the fly without loss of clock signals.
- Interface for the following modules:
  - PLL_IP—Three PLL_IP interfaces for each PLL on the IC.

— LPCG—Low-Power Clock Gating unit
— SRC—System Reset Controller
— GPC—Global Power Controller

7.1.2 CCM Blocks

The CCM includes the following submodules:

CCM_CLK_IGNITION	The purpose of this module it to manage the ignition process. This module becomes active after the CCM comes out of reset. It manages the ignition process by starting the Clock Amplifiers (CAMP), the Frequency Pre-Multiplier (FPM), and the PLLs. The ignition process ends when stable root clock outputs are available.
CCM_CLK_SWITCHER	This submodule receives the clock outputs of the three PLLs, together with the bypass clocks for three PLLs, and generates three clock outputs (pll1_sw_clk, pll2_sw_clk, pll3_sw_clk) for the ccm_clk_root_gen sub module. The submodule can route any of the PLL inputs to any of the three outputs.
CCM_CLK_ROOT_GEN	This sub module receives the three main clocks (pll1_sw_clk, pll2_sw_clk, pll3_sw_clk) and generates the output root clocks. The module also includes the programable clock dividers.
CCM_CLK_LOGIC	This sub-module generates the clock enables. It generates the clock enable signals based on data from CCM_LPM and CCM_IP. The clock enables are used by the Low-Power Clock Gating (LPCG) module to gate distributed clocks on and off.
CCM_LPM	Manages the low-power modes of the IC and module handshaking associate with low-power operation.
CCM_REGS	Manages the CCM memory map containing the programable registers and the connectivity to the IP bus. This module is connected to all the sub modules that need definition of programmable bits.
CCM_CLK_SRC_DIV	This sub-module directs various internal clocks which can be selected as i.MX51 outputs (through IOMUX), CLKO and CLKO2, used for observation and debug testing. These clocks are not optimized and therefore may introduce a jitter and are not suitable for supplying a clock signal to peripheral devices.
CCM_HND_SK	Manages the handshaking for those root clock dividers that require handshaking and manages the frequency changes during DVFS events.

Figure 7-1 shows the submodules contained in the CCM.

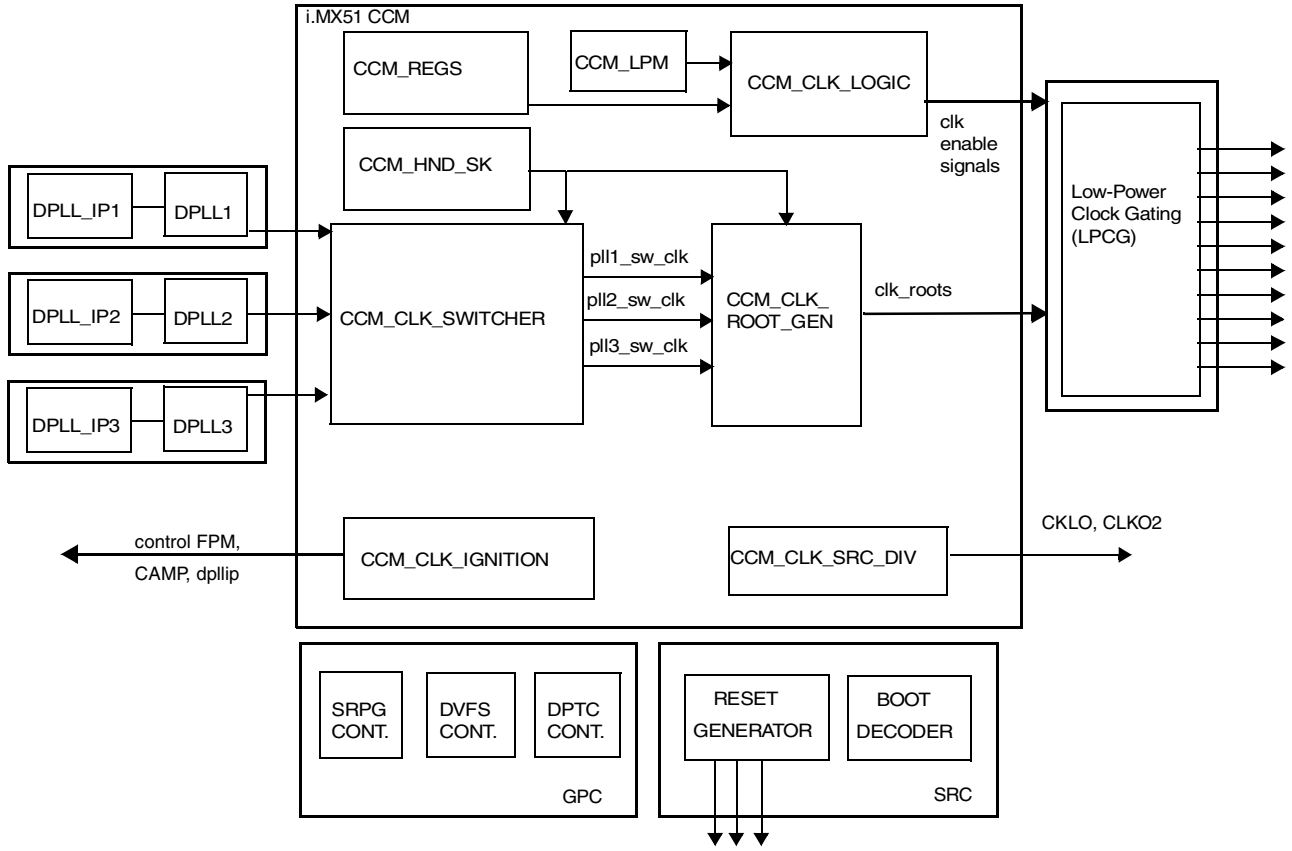


Figure 7-1. i.MX51 CCM Block Diagram

7.2 Detailed Signal Descriptions

7.2.1 External Signals Description

Table 7-1 describes the input/output signals of CCM module:

Table 7-1. CCM External Signals Description

Signal	I/O	Description
clko signals		
ipp_do_clko1	O	Clock observability 1 output
ccm_ipp_obe_clko1	O	BGA Contact output enable for Clock observability 1 output
ipp_do_clko2	O	Clock observability 2 output
ccm_ipp_obe_clko2	O	BGA Contact output enable for Clock observability 2 output
dptc_core1_clk_clko	I	DPTC core clock for observability
dptc_peripheral1_clk_clko	I	DPTC peripheral clock for observability

Table 7-1. CCM External Signals Description (continued)

Signal	I/O	Description
obs_output_0	O	Observability output
obs_output_1	O	Observability output
obs_output_2	O	Observability output
usbphy_pll_out_480	I	480MHZ USB PHY PLL clock for observability
General purpose signals		
cgpr_dout[31:0]	O	General purpose outputs
Input clocks from PADs		
ipp_di_clk	I	Reserved
ipp_ind_ckil	I	CKIL clock input
ipp_ind_clkss	I	Selects DPLL reference clock during reset
PMIC voltage control signals		
pmic_vstby_req	O	Goes to PMIC_VSTBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage.
pmic_vfuncional_ready	I	Signal coming from PMIC to indicate that the voltage started to change as result of change in pmic_vstby_req
FPM clock		
fpm_clk	I	FPM clock input
PLL signals		
pll1_main_clk	I	PLL1 clock
pll2_main_clk	I	PLL2 clock
pll3_main_clk	I	PLL3 clock
pll1_reference_clk	I	PLL reference clock
pll1_bypass_clk	I	PLL1 bypass clock
pll2_bypass_clk	I	PLL2 bypass clock
pll3_bypass_clk	I	PLL3 bypass clock
pll_bypass_en1	I	Enable bypass of PLL 1 clock
pll_bypass_en2	I	Enable bypass of PLL 2 clock
pll_bypass_en3	I	Enable bypass of PLL 3 clock
pll_lvs	O	Goes to LVS input in DPLLIPs (switches frequencies)
pll_lrf_sticky1	I	Asserts when PLL1 output is stable (lock ready flag)
pll_lrf_sticky2	I	Asserts when PLL2 output is stable (lock ready flag)
pll_lrf_sticky3	I	Asserts when PLL3 output is stable (lock ready flag)
Ignition signals		

Table 7-1. CCM External Signals Description (continued)

Signal	I/O	Description
fpm_lrf	I	FPM output is stable (lock ready flag)
ccm_fpm_en	O	Enable FPM
fpm_mult	O	0 - FPM multiply by 512 1 - FPM multiply by 1024
osc_clk	I	Internal oscillator clock input
cosc_en	O	Enable internal oscillator
cosc_pwrdown	O	Power down internal oscillator
ckih_CAMP1_clk	I	Input from the CKIH CAMP1
ckih2_CAMP2_clk	I	Input from the CKIH CAMP2
ccm_CAMP1_dis	O	Disable CKIH CAMP1
ccm_CAMP2_dis	O	Disable CKIH2 CAMP2
ref_clk_en_dpllip	O	Enable reference clock for DPLL
dpll_en_dpllip	O	Enable DPLLs
src_clock_ready	O	Notifies reset controller that the root clocks are ready
ccm_ref_en_b	O	Enable external reference clock (CKIH)

7.3 Memory Map and Register Definition

7.3.1 Memory Map

See the system memory map for the complete listing of memory map offset addresses. Table 7-3 shows the CCM memory map.

Table 7-3. CCM Memory Map

Base Address Offset (Register Abbreviation)	Register	Access	Reset Value	Section/Page
73FD_4000 (CCR)	CCM Control Register (CCR)	R/W	0x0000_xx00	7.3.3.1/7-14
73FD_4004 (CCDR)	CCM Control Divider Register (CCDR)	R/W	0x0000_0000	7.3.3.2/7-15
73FD_4008 (CSR)	CCM Status Register (CSR)	R/O	0x0000_0010	7.3.3.3/7-17
73FD_400c (CCSR)	CCM Clock Switcher Register (CCSR)	R/W	0x0000_0000	7.3.3.4/7-18
73FD_4010 (CACRR)	CCM Arm Clock Root Register (CACRR)	R/W	0x0000_0000	7.3.3.5/7-20
73FD_4014 (CBCDR)	CCM Bus Clock Divider Register(CBCDR)	R/W	0x1923_9145	7.3.3.6/7-21
73FD_4018 (CBCMR)	CCM Bus Clock Multiplexer Register (CBCMR)	R/W	0x0000_20C0	7.3.3.7/7-24

Table 7-3. CCM Memory Map (continued)

Base Address Offset (Register Abbreviation)	Register	Access	Reset Value	Section/Page
73FD_401c (CSCMR1)	CCM Serial Clock Multiplexer Register 1 (CSCMR1)	R/W	0xA6A2_A020	7.3.3.8/7-26
73FD_4020 (CSCMR2)	CCM Serial Clock Multiplexer Register 2 (CSCMR2)	R/W	0x02A5_A88A	7.3.3.9/7-29
73FD_4024 (CSCDR1)	CCM Serial Clock Divider Register 1 (CSCDR1)	R/W	0x00C3_0318	7.3.3.10/7-30
73FD_4028 (CS1CDR)	CCM SSI1 Clock Divider Register(CS1CDR)	R/W	0x0086_0041	7.3.3.11/7-34
73FD_402c (CS2CDR)	CCM SSI2 Clock Divider Register(CS2CDR)	R/W	0x0086_0041	7.3.3.12/7-35
73FD_4030 (CDCDR)	CCM DI Clock Divider Register(CDCDR)	R/W	0x0432_0DD2	7.3.3.13/7-37
73FD_4038 (CSCDR2)	CCM Serial Clock Divider Register 2(CSCDR2)	R/W	0x0209_0241	7.3.3.14/7-39
73FD_403c (CSCDR3)	CCM Serial Clock Divider Register 3(CSCDR3)	R/W	0x0001_0241	7.3.3.15/7-40
73FD_4040 (CSCDR4)	CCM Serial Clock Divider Register 4(CSCDR4)	R/W	0x0001_0241	7.3.3.16/7-42
73FD_4044 (CWDR)	CCM Wakeup Detector Register(CWDR)	R/W	0x0000_0000	7.3.3.17/7-43
73FD_4048 (CDHIPR)	CCM Divider Handshake In-Process Register(CDHIPR)	R/O	0x0000_0000	7.3.3.18/7-45
73FD_404c (CDCR)	CCM DVFS Control Register(CDCR)	R/W	0x0000_0001	7.3.3.19/7-48
73FD_4050 (CTOR)	CCM Testing Observability Register (CTOR)	R/W	0x0000_0000	7.3.3.20/7-49
73FD_4054 (CLPCR)	CCM Low Power Control Register(CLPCR)	R/W	0x0000_0079	7.3.3.21/7-52
73FD_4058 (CISR)	CCM Interrupt Status Register(CISR)	W1C	0x0000_0000	7.3.3.22/7-55
73FD_405c (CIMR)	CCM Interrupt Mask Register(CIMR)	R/W	0xFFFF_FFFF	7.3.3.23/7-58
73FD_4060 (CCOSR)	CCM Clock Output Source Register (CCOSR)	R/W	0x000A_0001	7.3.3.24/7-61
73FD_4064 (CGPR)	CCM General Purpose Register(CGPR)	R/W	0x0000_FE62	7.3.3.25/7-64
73FD_4068 (CCGR0)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_406c (CCGR1)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_4070 (CCGR2)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_4074 (CCGR3)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_4078 (CCGR4)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_407c (CCGR5)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_4080 (CCGR6)	CCM Clock Gating Register(CCGR)	R/W	0xFFFF_FFFF	7.3.3.26/7-66
73FD_4084 (CMEOR)	CCM Module Enable Override Register(CMEOR)	R/W	0xFFFF_FFFF	7.3.4.1/7-74

7.3.2 Register Summary

The conventions in Figure 7-2 and Table 7-4 serve as a key for the register summary and individual register diagrams.

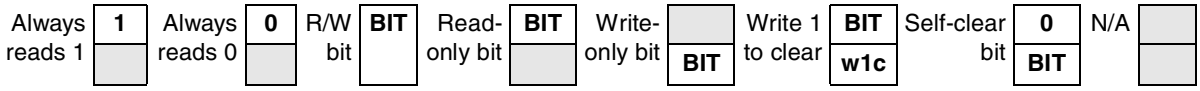


Figure 7-2. Key to Register Fields

Table 7-4 provides a key for register figures and tables and the register summary.

Table 7-4. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 7-5 shows the register summary for the CCM.

Table 7-5. i.MX51 CCM Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_4000 (CCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	cosc_en	fpm_mult	camp2_en	camp1_en	fpm_en	oscnt							
	W																

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_4004 (CCDR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ipu_hs_mask	emi_hs_mask
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
73FD_4008 (CSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	cosc_ready	lvs_value	camp2_ready	camp1_ready	fpm_ready	ref_en_b
	W																
73FD_400c (CCSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	lp_apm	step_sel[1:0]		pll2_div_podf[1:0]		pll3_div_podf[1:0]		pll1_sw_clk_sel	pll2_sw_clk_sel	pll3_sw_clk_sel
	W																
73FD_4010 (CACRR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0		0		0	ARM_podf [2:0]		
	W																
73FD_4014 (CBCDR)	R	0	ddr_high_freq_clk_sel	ddr_clk_podf			emi_clk_sel	periph_clk_sel	emi_slow_podf			axi_b_podf			axi_a_podf		
	W																
	R	nfc_podf			ahb_podf [2:0]			ipg_podf [1:0]		perclk_pred1[1:0]		perclk_pred2 [2:0]			perclk_podf [2:0]		
	W																
73FD_4018 (CBCMR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	gpu2d_clk_sel [1:0]	
	W																
	R	vpu_axi_clk_sel [1:0]		periph_apm_sel[1:0]		ddr_clk_sel [1:0]		arm_axi_clk_sel [1:0]		ipu_hsp_clk_sel [1:0]		gpu_clk_sel [1:0]		debug_apb_clk_sel [1:0]		perclk_lp_apm_sel	perclk_ipg_sel
	W																

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_401c (CSCMR1)	R	ssi_ext2_clk_sel [1:0]		ssi_ext1_clk_sel [1:0]		0	usb_phy_clk_sel	uart_clk_sel [1:0]		usboh3_clk_sel [1:0]		esdhc1_clk_sel [1:0]		esdhc3_clk_sel	esdhc4_clk_sel	esdhc2_clk_sel [1:0]	
	W																
	R	ssi1_clk_sel [1:0]		ssi2_clk_sel [1:0]		ssi3_clk_sel	vpu_rclk_sel	ssi_apm_clk_sel		tve_clk_sel	tve_ext_clk_sel	cspi_clk_sel [1:0]		spdif_xtal_clk_sel		ssi_ext2_com	ssi_ext1_com
	W																
73FD_4020 (CSCMR2)	R	di1_clk_sel[1:0]			di0_clk_sel[1:0]			csi_mclk2_clk_sel[1:0]		csi_mclk1_clk_sel[1:0]		0	0	0	0	0	0
	W																
	R	hsi2c_clk_sel [1:0]		firi_clk_sel [1:0]		sim_clk_sel [1:0]		0	0	0	0	spdif1_com	spdif0_com	spdif1_clk_sel [1:0]		spdif0_clk_sel [1:0]	
	W																
73FD_4024 (CSCDR1)	R	0	0	0	0	0	0	0	esdhc2_clk_pred [2:0]			esdhc2_clk_podf[2:0]			esdhc1_mshc1_clk_pred [2:0]		
	W																
	R	pgc_clk_podf[1:0]		esdhc1_mshc1_clk_podf[2:0]			usboh3_clk_pred[2:0]			usboh3_clk_podf[1:0]		uart_clk_pred[2:0]			uart_clk_podf[2:0]		
	W																
73FD_4028 (CS1CDR)	R	0	0	0	0	0	0	0	ssi_ext1_clk_pred [2:0]			ssi_ext1_clk_podf [5:0]					
	W																
	R	0	0	0	0	0	0	0	ssi1_clk_pred [2:0]			ss1_clk_podf [5:0]					
	W																
73FD_402c (CS2CDR)	R	0	0	0	0	0	0	0	ssi_ext2_clk_pred [2:0]			ssi_ext2_clk_podf [5:0]					
	W																
	R	0	0	0	0	0	0	0	ssi2_clk_pred [2:0]			ss2_clk_podf [5:0]					
	W																
73FD_4030 (CDCDR)	R	0	tve_clk_pred [2:0]			spdif0_clk_pred [2:0]			spdif0_clk_podf [5:0]						spdif1_clk_pred [2:0]		
	W																
	R	0	spdif1_clk_podf [5:0]						di_clk_pred [2:0]			usb_phy_pred[2:0]			usb_phy_podf[2:0]		
	W																
73FD_4038 (CSCDR2)	R	0	0	0	0	ecspi_clk_pred[2:0]			ecspi_clk_podf[5:0]						sim_clk_pred[2:0]		
	W																
	R	0	sim_clk_podf[5:0]						0	0	0	0	0	0	0	0	0
	W																

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_403c (CSCDR3)	R	0	0	0	0	0	**0**	**0**	0	**0**	**0**	**0**	**0**	**0**	hsi2c_clk_pred[2:0]		
	W																
	R	0	hsi2c_clk_podf[5:0]						firi_clk_pred[2:0]			firi_clk_podf[5:0]					
	W																
73FD_4040 (CSCDR4)	R	0	0	0	0	0	**0**	**0**	0	**0**	**0**	**0**	**0**	**0**	csi_mclk2_clk_pred[2:0]		
	W																
	R	0	csi_mclk2_clk_podf[5:0]						csi_mclk1_clk_pred[2:0]			csi_mclk1_clk_podf[5:0]					
	W																
73FD_4044 (CWDR)	R	0	0	0	0	0	0	0	0	0	0	gpio1_9_dir	gpio1_8_dir	gpio1_7_dir	gpio1_6_dir	gpio1_5_dir	gpio1_4_dir
	W																
	R	0	0	0	0	gpio1_9_icr		gpio1_8_icr		gpio1_7_icr		gpio1_6_icr		gpio1_5_icr		gpio1_4_icr	
	W																
73FD_4048 (CDHIPR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	arm_podf_busy
	W																
	R	0	0	0	0	0	0	0	ddr_high_freq_clk_sel_busy	ddr_podf_busy	emi_clk_sel_busy	periph_clk_sel_busy	nfc_podf_busy	ahb_podf_busy	emi_slow_podf_busy	axi_b_podf_busy	axi_a_podf_busy
	W																
73FD_404c (CDCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	sw_periph_clk_div_req_status	sw_periph_clk_div_req	software_DVFS_en	0	0	arm_freq_shift_divider	periph_clk_DVFS_podf[1:0]	
	W									w1c							

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_4050 (CTOR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	obs_en	obs_spare_output_0_sel					obs_spare_output_1_sel				obs_spare_output_2_sel			
	W																
73FD_4054 (CLPCR)	R	0	0	0	0	0	0	0	0	0	bypass_scc_lpm_hs	bypass_max_lpm_hs	bypass_sdma_lpm_hs	bypass_emi_lpm_hs	bypass_ipu_lpm_hs	bypass_rtic_lpm_hs	bypass_sahara_lpm_hs
	W																
	R	0	0	0	apm_sdma_clk_gate_en_bit	cosc_pwrdown	stby_count		VSTBY	dis_ref_osc	SBYOS	ARM_clk_dis_on_lpm	lpsr_clk_sel		bypass_pmic_vfunctional_ready	LPM[1:0]	
	W																
73FD_4058 (CISR)	R	0	0	0	0	0	arm_podf_loaded	ddr_high_freq_clk_sel_loaded	ddr_clk_podf_loaded	emi_clk_sel_loaded	periph_clk_sel_loaded	nfc_podf_loaded	ahb_podf_loaded	emi_slow_podf_loaded	axi_b_podf_loaded	axi_a_podf_loaded	dividers_loaded
	W						w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	gpio1_9_wakeup_det	gpio1_8_wakeup_det	gpio1_7_wakeup_det	gpio1_6_wakeup_det	gpio1_5_wakeup_det	gpio1_4_wakeup_det	sdhc2_wakeup_det	sdhc1_wakeup_det	kpp_wakeup_det	cosc_ready	CAMP2_ready	CAMP1_ready	fpm_ready	lrf_pll3	lrf_pll2	lrf_pll1
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_405c (CIMR)	R	1	1	1	1	1	mask_arm_podf_loaded	mask_ddr_high_freq_clk_sel_loaded	mask_ddr_podf_loaded	mask_emi_clk_sel_loaded	mask_periph_clk_sel_loaded	mask_nfc_podf_loaded	mask_ahb_podf_loaded	mask_emi_slow_podf_loaded	mask_axi_b_podf_loaded	mask_axi_a_podf_loaded	mask_dividers_loaded
	W																
	R	mask_gpio1_9_wakeup_det	mask_gpio1_8_wakeup_det	mask_gpio1_7_wakeup_det	mask_gpio1_6_wakeup_det	mask_gpio1_5_wakeup_det	mask_gpio1_4_wakeup_det	mask_sdhc2_wakeup_det	mask_sdhc1_wakeup_det	mask_kpp_wakeup_det	mask_cosc_ready	mask_CAMP2_ready	mask_CAMP1_ready	mask_fpm_ready	mask_lrf_pll3	mask_lrf_pll2	mask_lrf_pll1
	W																
73FD_4060 (CCOSR)	R	0	0	0	0	0	0	0	cko2_en	cko2_div[2:0]			cko2_sel[4:0]				
	W																
	R	0	0	0	0	0	0	0	0	cko1_en	cko1_div[2:0]			cko1_sel[3:0]			
	W																
73FD_4064 (CGPR)	R	0	0	0	0	0	0	0	ARM_clk_input_sel	arm_async_ref_en	arm_async_ref_sel						
	W																
	R	arm_async_ref_sel[5]	1	1	1	1	1	1	0	0	1	overide_apm_emi_int1_clock_gating	efuse_prog_supply_gate	fpm_mux_select	0	1	0
	W																
73FD_4068 (CCGR0)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_406c (CCGR1)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_4070 (CCGR2)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																

Table 7-5. i.MX51 CCM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
73FD_4074 (CCGR3)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_4078 (CCGR4)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_407c (CCGR5)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_4080 (CCGR6)	R	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
	W																
	R	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
	W																
73FD_4084 (CMEOR)	R	1	1	1	1	1	0	0	0	mod_en_ov_emi_m7	mod_en_ov_emi_m6	mod_en_ov_emi_m5	mod_en_ov_emi_m4	mod_en_ov_emi_m3	mod_en_ovl_emi_m2	mod_en_ovl_emi_m1	mod_en_ov_emi_m0
	W																
	R	mod_en_ov_emi_int1	mod_en_ov_emi_slow	mod_en_ov_emi_fast	mod_en_ov_emi_garb	1	mod_en_ov_gpu2d	mod_en_ov_vpu	mod_en_ov_dap	mod_en_ov_gpu	mod_en_ov_epit	mod_en_ov_gpt	mod_en_ov_esdhc	mod_en_ov_iim	mod_en_ov_owire	1	mod_en_ov_sahara
	W																

7.3.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number.

7.3.3.1 CCM Control Register (CCR)

Figure 7-3 represents the CCM Control Register (CCR), which contains bits to control general operation of CCM. Table 7-6 provides its field descriptions.

73FD_4000 (CCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	cosc_ en	fpm_ mult	camp 2_en	camp 1_en	fpm_ en	oscnt							
W																
Reset	0	0	0	—	1	1	1	—	1	1	1	1	1	1	1	1

Figure 7-3. CCM Control Register (CCR)

Table 7-6. CCR Field Descriptions

Field	Description
31–13	Reserved
12 cosc_en	On-chip oscillator enable bit - this bit value is reflected on the output cosc_en. The ignition process sets this bit if the on-chip oscillator is selected as the clock source. The reset value of this bit is determined during the ignition process based on CLKSS input. In Run mode, the software controls the on-chip oscillator enable/disable through this bit. If this bit is changed from '0' to '1' then the CCM enables the on-chip oscillator and after counting the **oscnt CKIL** clock cycles it indicates the on-chip oscillator is ready by an interrupt **cosc_ready** and by the **cosc_ready** status bit. **Note:** The cosc_en bit should only be changed if the on-chip oscillator is not chosen as the clock source by the DPLL_IPs. 0 Disable the on-chip oscillator 1 Enable the on-chip oscillator
11 FPM_MULT	fpm_mult - controls the multiplicand of FPM. This bit can be changed only in case FPM is not chosen as the source of clock generation. FPM should be disabled (FPM_en=0) when changing this value. **Note:** It multiplies the CKIL (32Khz) clock. 0 FPM multiply by 512 1 FPM multiply by 1024
10 CAMP2_EN	CAMP2 Enable bit - the ignition process always setS this bit. In run mode, software can control CAMP2 enable/disable using this bit. If this bit is changed from '0' to '1' then the CCM enables the CAMP2 and after counting oscnt CKIL's it will notify that CAMP2 is ready by the CAMP2_ready interrupt (if the interrupt is not masked) and by the CAMP2_ready status bit. **Note:** The CAMP2_en bit should only be changed when CAMP2 is not chosen as a clock source. 0 disable CAMP2 1 enable CAMP2 Note: CCM has an output ccm_CAMP2_dis. this signal is inverted from the CAMP2_en bit, for example, when CAMP2_en bit is '1' the ccm_CAMP2_dis signal is '0'.

Table 7-6. CCR Field Descriptions (continued)

Field	Description
9 CAMP1_EN	CAMP1 Enable bit - the ignition process will always set this bit. In run mode, software can control CAMP1 enable/disable through this bit. If this bit is changed from '0' to '1' then CCM will enable the CAMP1 and after counting oscnt CKIL's it will notify that CAMP1 is ready by a interrupt CAMP1_ready and by status bit CAMP1_ready. The CAMP1_en bit should be changed only when CAMP1 is not chosen as a clock source. 0 disable CAMP1 1 enable CAMP1 Note: CCM has an output ccm_CAMP1_dis. this signal will be inverted from the CAMP1_en bit, that is, when CAMP1_en bit is '1' the ccm_CAMP1_dis signal is '0'.
8 FPM_EN	FPM Enable bit - the ignition process will set this bit if the FPM is chosen as the clock source, that is, the reset value of this bit is chosen from the ignition process based on CLKSS input. In run mode, software can control FPM enable/disable through this bit. If this bit is changed from '0' to '1' then CCM will enable FPM and after FPM lock ready flag is asserted it will notify that FPM is ready by a interrupt FPM_ready and by status bit FPM_ready. **Note:** The FPM_en bit should be changed only when FPM is not chosen as the clock source, that is, the DPLL_ip's are not choosing the FPM as the clock source. 0 disable FPM 1 enable FPM
7–0 OSCNT	Oscillator ready counter value. These bits define value of 32 KHz counter that serves as the counter for oscillator lock time. This is used for both the on-chip oscillator lock time and the time that CAMP1 and CAMP2 will be ready since the CAMPs receives the external oscillator clock. Current estimation is ~5ms. This counter will be used in ignition sequence and in wake from STOP sequence if sbyos bit was set , to notify that the on-chip oscillator output is ready for the DPLL_ip to use and only then the gate in DPLL_ip is enabled. 00000000 count 1 CKIL 11111111 count 256 CKIL's (Default)

7.3.3.2 CCM Control Divider Register (CCDR)

Figure 7-4 represents the CCM Control Divider Register (CCDR). The only function of this register is to control the masking of handshaking with some of the dividers.

Table 7-7 provides its field descriptions.

73FD_4004 (CCDR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ipu_hs_mask	emi_hs_mask
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-4. CCM Control Divider Register (CCDR)

NOTE

The bits in the CCDR register can only be changed after a completion of the handshake.

Table 7-7. CCDR Field Descriptions

Field	Description
31–19	Reserved
18	Reserved. **Note:** Functionality is no longer supported. Bit must be set for better power saving - along with configuration of CLPCR[23] and CCGR4[13:6] bits. Please refer to description of these bits for the recommended values.
17 ipu_hs_mask	During load_dividers procedure[1] this bit allows or masks the handshake with IPU module. 0 allow handshake with IPU module 1 mask handshake with IPU module
16 emi_hs_mask	During load_dividers procedure[1] this bit allows or masks the handshake with EMI module. 0 allow handshake with EMI module 1 mask handshake with EMI module
15–0	Reserved

7.3.3.3 CCM Status Register (CSR)

Figure 7-5 represents the CCM status Register (CSR). The status bits are read only bits. Table 7-8 provides its field descriptions.

73FD_4008 (CSR) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	cosc_ready	lvs_value	camp2_ready	camp1_ready	fpm_ready	ref_en_b
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Figure 7-5. CCM Status Register (CSR)

Table 7-8. CSR Field Descriptions

Field	Description
31–6	Reserved
5 cosc_ready	Status indication of on-chip oscillator. This bit is asserted if the on-chip oscillator is enabled and the on-chip oscillator is not powered down, and if the oscnt counter has finished counting. 0 - on-chip oscillator is not ready. 1- on-chip oscillator is ready.
4 lvs_value	Status of the value of pll_lvs output of the CCM. For more details about the pll_lvs see the DPLL IP chapter. 0 - value of pll_lvs output is ‘0’ 1- value of pll_lvs output is ‘1’
3 CAMP2_ready	Status indication of CAMP2. This is asserted if CAMP2 is enabled and if the oscnt counter has finished counting. 0 - CAMP2 is not ready. 1- CAMP2 is ready.
2 CAMP1_ready	Status indication of CAMP1. This is asserted if CAMP1 was enabled and if oscnt counter has finished counting. 0 - CAMP1 is not ready. 1- CAMP1 is ready.

Table 7-8. CSR Field Descriptions (continued)

Field	Description
1 fpm_ready	Status indication of FPM. This will be asserted if FPM was enabled and FPM notified by asserting its ready signal. 0 - FPM is not ready. 1- FPM is ready.
0 ref_en_b	Status of the value of ref_en_b output of ccm 0 - value of ref_en_b is '0' 1- value of ref_en_b is '1'

7.3.3.4 CCM Clock Switcher Register (CCSR)

Figure 7-6 represents the CCM Clock Switcher register (CCSR). The CCSR register contains bits to control the switcher sub module dividers and multiplexers. Table 7-9 provides its field descriptions.

73FD_400c (CCSR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	lp_apm	step_sel[1:0]		pll2_div_podf[1:0]		pll3_div_podf[1:0]		pll1_sw_cllk_sel	pll2_sw_cllk_sel	pll3_sw_clk_sel
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-6. CCM Clock Switch Register (CCSR)

Table 7-9. CCSR Field Descriptions

Field	Description
31–9	Reserved
9 lp_apm	Selects the option to be chosen for the Low-Power Audio Playback source clock. 0 On-chip oscillator clock output 1 FPM clock output

Table 7-9. CCSR Field Descriptions (continued)

Field	Description
8–7 step_sel [1:0]	Selects the option to be chosen for the step frequency when shifting ARM frequency. This bits controls the step_clk. 00 clock source 4 - source for lp_apm. (default) 01 pll1 bypass clock 10 divided pll2 clock 11 divided pll3 clock **Note:** The mux can only be switched while the output is not being used. **Note:** To conserve power, Freescale recommends this mux not be set to the pll2 and pll3 options when not being used. When PLL2 and PLL3 are running (but not selected), a small amount of power is until consumed, but it does not enter the glitchless mux and hence less clock tree power is consumed".
6–5 pll2_div_podf [1:0]	Divider for PLL2 clock. 00 divide by 1 01 divide by 2 10 divide by 3 11 divide by 4 **Note:** This field can only be changed during the period that its output is not used.
4–3 pll3_div_podf [1:0]	Divider for PLL3 clock. 00 divide by 1 01 divide by 2 10 divide by 3 11 divide by 4 **Note:** This field can only be changed during the period that its output is not used.
2 pll1_sw_clk_sel	Selects source to generate pll1_sw_clk. 0 pll1_main_clk (default) 1 step_clk **Note:** This bit is OR'd with the pll_bypass_en1 signal and DVFS_control signal. If one of the sources requests to move to step_clk (pll1_sw_clk=1, or pll_bypass_en1=1, or DVFS_control=1) then the source clock for pll1_sw_clk will be step_clk. Only if both sources request pll1_main_clk (pll1_sw_clk=0, and pll_bypass_en1=0 and DVFS_control=0) will the pll1_sw_clk source clock be pll1_main_clk.
1 pll2_sw_clk_sel	Selects source to generate pll2_sw_clk. This bit should only be used for testing purposes. 0 pll2_main_clk(Default) 1 pll2 bypass clock **Note:** This bit is OR'd with pll_bypass_en2 signal. If one of the sources requests to move to pll2 bypass clk (pll2_sw_clk=1 or pll_bypass_en2=1) then the pll2_sw_clk will be pll2 bypass clk. Only if both sources request pll2_main_clk (pll2_sw_clk=0 and pll_bypass_en2=0) then the pll2_sw_clk will be pll2_main_clk.
0 pll3_sw_clk_sel	Selects source to generate pll3_sw_clk. This bit should only be used for testing purposes. 0 pll3_main_clk(Default) 1 pll3 bypass clock Note: this bit is OR'd with pll_bypass_en3 signal. If one of the sources requests to move to pll3 bypass clk (pll3_sw_clk=1 or pll_bypass_en3=1) then the pll3_sw_clk will be pll3 bypass clk. Only if both sources request pll3_main_clk (pll3_sw_clk=0 and pll_bypass_en3=0) then the pll3_sw_clk will be pll3_main_clk.

7.3.3.5 CCM Arm Clock Root Register (CACRR)

Figure 7-7 represents the CCM Arm Clock Root register (CACRR). The CACRR register contains bits to control the ARM clock root generation. Table 7-10 provides its field descriptions.

73FD_4010 (CACRR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0		0		0	arm_podf [2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-7. CCM ARM Clock Root Register (CACRR)

Table 7-10. CACRR Field Descriptions

Field	Description
31–3	Reserved
2–0 arm_podf [2:0]	Divider for ARM clock root. Note: if arm_freq_shift_divider is set to ‘1’ then any new write to arm_podf will be held until arm_clk_switch_req signal is asserted. 000 divide by 1(default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8

7.3.3.6 CCM Bus Clock Divider Register(CBCDR)

Figure 7-8 represents the CCM Bus Clock Divider Register (CBCDR). The CBCDR register contains bits to control the clock generation sub module dividers. Table 7-11 provides its field descriptions.

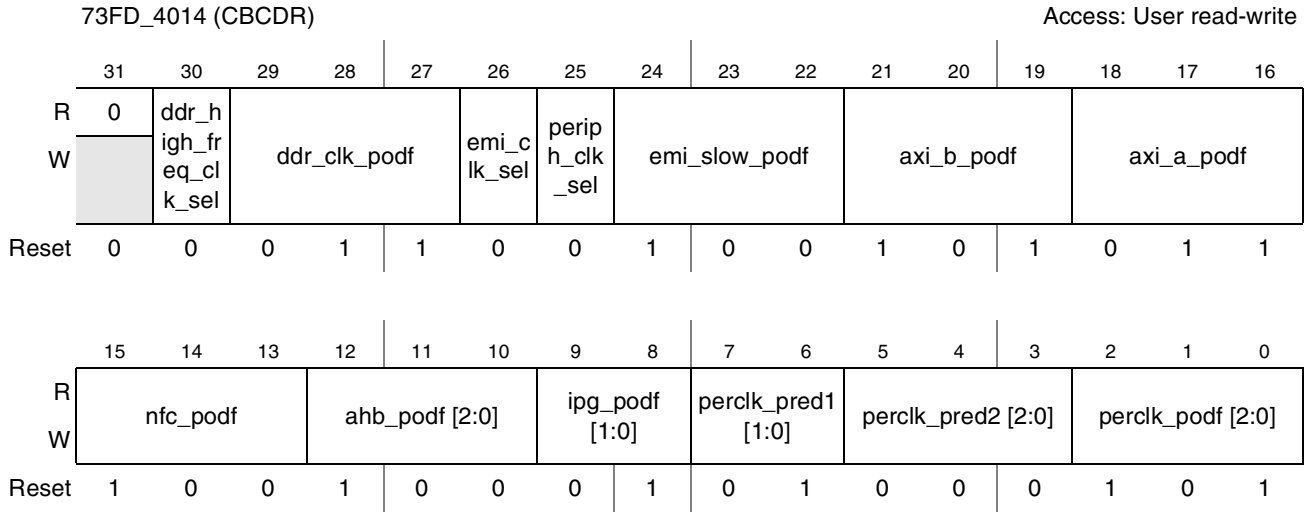


Figure 7-8. CCM Bus Clock Divider Register (CBCDR)

Table 7-11. CBCDR Field Descriptions

Field	Description
31	Reserved
30 ddr_high_freq_clk_sel	Selector for DDR main clock. 0 derive clock from DDR mux clock source. 1 derive clock from divided PLL1 clock source. **Note:** Any change of this multiplexer will involve handshake with EMI on the DDR part. The handshake can be masked by setting CCDR[16] bit. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.
29–27 ddr_clk_podf [2:0]	Divider for DDR podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 (default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI. The handshake can be masked by setting CCDR[16] bit. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details,
26 emi_clk_sel	Selector for EMI clock group 0 derive clock from DVFS divider 1 derive clock from AHB clock root **Note:** Any change of this multiplexer might involve handshake with EMI and IPU. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) register for the handshake busy bits.

Table 7-11. CBCDR Field Descriptions (continued)

Field	Description
25 periph_clk_sel	Selector for peripheral main clock. 0 derive clock from pll2_sw_clk clock source. 1 derive clock from periph_apm_clk clock source. **Note:** Any change of this multiplexer will involve handshake with EMI and IPU - a similar handshake to the one that is performed on peripheral DVFS. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR)
24–22 emi_slow_podf [2:0]	Divider for EMI slow podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5(default) 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI and IPU. See CDHIPR register for the handshake busy bits. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.
21–19 axi_b_podf [2:0]	Divider for axi b podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 (default) 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI and IPU. See CDHIPR register for the handshake busy bits. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.
18–16 axi_a_podf [2:0]	Divider for axi a podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 (default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI and IPU. See CDHIPR register for the handshake busy bits. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.

Table 7-11. CBCDR Field Descriptions (continued)

Field	Description
15–13 nfc_podf [2:0]	Divider for nfc podf. 000 restricted 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 (Default) 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI and IPU. See CDHIPR register for the handshake busy bits. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.
12–10 ahb_podf [2:0]	Divider for ahb podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 (Default) 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Any change of this divider might involve handshake with EMI and IPU. See CDHIPR register for the handshake busy bits. See Section 7.3.3.18, CCM Divider Handshake In-Process Register(CDHIPR) for details.
9 -8 ipg_podf [1:0]	Divider for ipg podf. 00 divide by 1 01 divide by 2 (Default) 10 divide by 3 11 divide by 4
7–6 perclk_pred1 [1:0]	Divider for perclk pred1 00 divide by 1 01 divide by 2 (default) 10 divide by 3 11 divide by 4 Note: Divider should be updated when output clock is gated.

Table 7-11. CBCDR Field Descriptions (continued)

Field	Description
5–3 perclk_pred2 [2:0]	Divider for perclk pred2. 000 divide by 1(default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
2–0 perclk_podf [2:0]	Divider for perclk podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6(default) 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.

7.3.3.7 CCM Bus Clock Multiplexer Register (CBCMR)

Figure 7-9 represents the CCM Bus Clock Multiplexer Register (CBCMR). The CBCMR register contains bits to control the multiplexers that generate the bus clocks. Table 7-12 provides its field descriptions.

73FD_4018 (CBCMR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	gpu2d_clk_sel [1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	vpu_axi_clk_sel [1:0]		periph_apm_sel[1:0]		ddr_clk_sel [1:0]		arm_axi_clk_sel [1:0]		ipu_hsp_clk_sel [1:0]		gpu_clk_sel [1:0]		debug_apb_clk_sel [1:0]		perclk_lp_apm_sel	perclk_ipg_sel
Reset	0	0	1	0	0	0	0	0	1	1	0	0	0	0	0	0

Figure 7-9. CCM Bus Clock Multiplexer Register (CBCMR)

Table 7-12. CBCMR Field Descriptions

Field	Description
31-18	Reserved
17-16 gpu2d_clk_sel [1:0]	Selector for gpu2d clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
15–14 vpu_axi_clk_sel [1:0]	Selector for VPU axi clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
13–12 periph_apm_sel [1:0]	Selector for peripheral clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll3_sw_clk 10 lp_apm clock (default) 11 reserved
11–10 ddr_clk_sel [1:0]	Selector for DDR clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
9–8 arm_axi_clk_sel [1:0]	Selector for ARM axi clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
7–6 ipu_hsp_clk_sel [1:0]	Selector for IPU hsp clock multiplexer 00 derive clock from axi a 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root (Default)
5–4 gpu_clk_sel [1:0]	Selector for GPU clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
3–2 debug_apb_clk_sel [1:0]	Selector for debug apb clock multiplexer 00 derive clock from axi a (Default) 01 derive clock from axi b 10 derive clock from emi_slow_clk_root 11 derive clock from ahb clock root
1 perclk_lp_apm_sel	Controls if the root clock generation of perclk will be from peripherals main clock or lp_apm source. 0 generate perclk_root from peripherals main clock source. 1 generate perclk_root from lp_apm source.
0 perclk_ipg_sel	Selector for perclk multiplexer - allows to select between ipg_clk or the division of chosen PLL. 0 select perclk generation from division of pll frequency 1 select perclk generation from ipg_clk

Note: Any change on the above multiplexer will have to be done while the module that its clock is affected is not functional and the respective clock is gated in LPCG. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

The change for arm_axi_clk_sel should be done through SDMA so that ARM will not use this clock during the change and the clock will be gated in LPCG.

7.3.3.8 CCM Serial Clock Multiplexer Register 1 (CSCMR1)

Figure 7-10 represents the CCM Serial Clock Multiplexer Register 1 (CSCMR1). The CSCMR1 register contains bits to control the multiplexers that generate the serial clocks. Table 7-13 provides its field descriptions.

73FD_401c (CSCMR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ssi_ext2_clk_sel [1:0]		ssi_ext1_clk_sel [1:0]		0	usb_phy_clk_sel	uart_clk_sel [1:0]		usboh3_clk_sel [1:0]		esdhc1_clk_sel [1:0]		esdhc3_clk_sel	esdhc4_clk_sel	esdhc2_clk_sel [1:0]	
Reset	1	0	1	0	0	1	1	0	1	0	1	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ssi1_clk_sel [1:0]		ssi2_clk_sel [1:0]		ssi3_clk_sel	vpu_rclk_sel	ssi_apm_clk_sel		tve_clk_sel	tve_ext_clk_sel	cspi_clk_sel [1:0]		spdif_xtal_clk_sel		ssi_ext2_com	ssi_ext1_com
Reset	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 7-10. CCM Serial Clock Multiplexer Register 1 (CSCMR1)

Table 7-13. CSCMR1 Field Descriptions

Field	Description
31–30 ssi_ext2_clk_sel [1:0]	Selector for ssi_ext2 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk (default) 11 derive clock from ssi_lp_apm_clk
29–28 ssi_ext1_clk_sel [1:0]	Selector for ssi_ext1 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 derive clock from ssi_lp_apm_clk
27	Reserved.
26 usb_phy_clk_sel	Selector for usb_phy clock multiplexer 0 derive clock from oscillator 1 derive clock from divided output of PLL3 (Default)

Table 7-13. CSCMR1 Field Descriptions (continued)

Field	Description
25–24 uart_clk_sel[1:0]	Selector for UART clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
23–22 usboh3_clk_sel [1:0]	Selector for USBOH3 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
21–20 esdhc1_clk_sel [1:0]	Selector for ESDHC1 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
19 esdhc3_clk_sel	Selector for ESDHC3 clock multiplexer 0 derive clock from esdhc1_clk_sel(Default) 1 derive clock from esdhc2_clk_sel
18 esdhc4_clk_sel	Selector for ESDHC4 clock multiplexer 0 derive clock from esdhc1_clk_sel(Default) 1 derive clock from esdhc2_clk_sel
17–16 esdhc2_clk_sel [1:0]	Selector for ESDHC2 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
15–14 ssi1_clk_sel [1:0]	Selector for SSI1 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 derive clock from ssi_lp_apm_clk
13–12 ssi2_clk_sel [1:0]	Selector for SSI2 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk (default) 11 derive clock from ssi_lp_apm_clk
11 ssi3_clk_sel	Selector for SSI3 clock multiplexer 0 derive clock from ssi1_clk_root(Default) 1 derive clock from ssi2_clk_root
10 vpu_rclk_sel	Selector for vpu_rclk clock multiplexer 0 derive clock from osc_clk (default) 1 derive clock from ckih_CAMP1_clk
9–8 ssi_apm_clk_sel	Controls the multiplexer for the ssi_lp_apm_clk clock generation. 00 CAMP1 of CKIH (default) 01 LP-APM clock selector output 10 CAMP2 of CKIH2 11 Reserved

Table 7-13. CSCMR1 Field Descriptions (continued)

Field	Description
7 tve_clk_sel	Controls the multiplexer for the TVE clock generation. 0 PLL3 divided clock 1 External clock source (either osc or CAMP1)
6 tve_ext_clk_sel	Controls the multiplexer for the TVE external clock input. 0 Oscillator output 1 CKIh through CAMP1
5–4 ecspi_clk_sel [1:0]	Selector for ECSPI clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
3–2 spdif_xtal_clk_sel	Controls the multiplexer for the spdif_xtal_clk clock generation. 00 Oscillator output (default) 01 CAMP1 of CKIH 10 CAMP2 of CKIH2 11 reserved
1 ssi_ext2_com	Controls the multiplexer to communize ssi_ext2 clock generation based on ssi2 clock root. 0 generate ssi_ext2_clk_root from dividers 1 generate ssi_ext2_clk_root from ssi2_clk_root.
0 ssi_ext1_com	Controls the multiplexer to communize ssi_ext1 clock generation based on ssi1 clock root. 0 generate ssi_ext1_clk_root from dividers 1 generate ssi_ext1_clk_root from ssi1_clk_root.

NOTE

Before any change can be made to the multiplexor ensure the clock being switched is not being used and is gated off. Switching the clocks while they are in use can produce unexpected results.

7.3.3.9 CCM Serial Clock Multiplexer Register 2 (CSCMR2)

Figure 7-11 represents the CCM Serial Clock Multiplexer Register 2 (CSCMR2). The CSCMR2 register contains bits to control the multiplexers that generate the serial clocks. Table 7-14 provides its field descriptions.

73FD_4020 (CSCMR2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	di1_clk_sel[1:0]			di0_clk_sel[1:0]			csi_mclk2_clk_sel[1:0]		csi_mclk1_clk_sel[1:0]		0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	hsi2c_clk_sel [1:0]		firi_clk_sel [1:0]		sim_clk_sel [1:0]		0	0	0	0	spdif1_com	spdif0_com	spdif1_clk_sel [1:0		spdif0_clk_sel [1:0]	
W																
Reset	1	0	1	0	1	0	0	0	1	0	0	0	1	0	1	0

Figure 7-11. CCM Serial Clock Multiplexer Register 2 (CSCMR2)

Table 7-14. CSCMR2 Field Descriptions

Field	Description
31–29 di1_clk_sel [2:0]	Selector for DI1 clock multiplexer 000 derive clock from divided pll3. (Default) 001 derive clock from oscillator 010 derive clock from ckih CAMP1 clock. 011 derive clock from tve_di_clock - in this case TVE will supply the di clock. See the TVE chapter for details. 100 derive clock from ipp_di1_clk - clock source will be generated from the IOMUX. 101 - 111 - reserved.
28–26 di0_clk_sel [2:0]	Selector for di0 clock multiplexer 000 derive clock from divided pll3. (Default) 001 derive clock from oscillator 010 derive clock from ckih CAMP1 clock. 011 derive clock from tve_di_clock - in this case tve will supply the di clock. See the TVE chapter for details. 100 derive clock from ipp_di0_clk - clock source will be generated from the IOMUX. 101 - 111 - reserved.
25–16	Reserved
15–14 hsi2c_clk_sel [1:0]	Selector for hsi2c clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock

Table 7-14. CSCMR2 Field Descriptions (continued)

Field	Description
13–12 firi_clk_sel [1:0]	Selector for FIRI clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
11–10 sim_clk_sel [1:0]	Selector for sim clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 lp_apm clock
9–4	Reserved
5 spdif1_com	Controlls the multiplexer configuration for spdif1 clock generation based on ssi2 clock root. 0 generate spdif1_clk_root from dividers 1 generate spdif1_clk_root from ssi2_clk_root.
4 spdif0_com	Controlls the multiplexer to configuration for spdif0 clock generation based on ssi1 clock root. 0 generate spdif0_clk_root from dividers 1 generate spdif0_clk_root from ssi1_clk_root.
3–2 spdif1_clk_sel [1:0]	Selector for spdif1 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 spdif_xtal_clk clock
1–0 spdif0_clk_sel [1:0]	Selector for spdif0 clock multiplexer 00 derive clock from pll1_sw_clk 01 derive clock from pll2_sw_clk 10 derive clock from pll3_sw_clk(Default) 11 spdif_xtal_clk clock

NOTE

Before any change can be made to the multiplexor ensure the clock being switched is not being used and is gated off. Switching the clocks while they are in use can produce unexpected results.

7.3.3.10 CCM Serial Clock Divider Register 1 (CSCDR1)

Figure 7-12 represents the CCM Serial Clock Divider Register 1 (CSCDR1). The CSCDR1 register contains bits to control the clock generation sub module dividers. Table 7-15 provides its field descriptions.

73FD_4024 (CSCDR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	esdhc2_clk_pred [2:0]			esdhc2_clk_podf[2:0]			esdhc1_clk_pred [2:0]		
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	pgc_clk_podf [1:0]		esdhc1_clk_podf[2:0]			usboh3_clk_pred[2:0]			usboh3_clk_podf[1:0]		uart_clk_pred[2:0]			uart_clk_podf[2:0]		
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	1	1	0	0	0

Figure 7-12. CCM Serial Clock Divider Register 1(CSCDR1)

Table 7-15. CSCDR1 Field Descriptions

Field	Description
31–25	Reserved
24–22 esdhc2_clk_pred [2:0]	Divider for esdhc2 clock pred. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4(Default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
21–19 esdhc2_clk_podf[2:0]	Divider for esdhc2 clock podf. 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
18–16 esdhc1_clk_pred [2:0]	Divider for esdhc1 clock pred. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4(Default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.

Table 7-15. CSCDR1 Field Descriptions (continued)

Field	Description
15–14 pgc_clk_podf[1:0]	Divider for pgc (power gating controller) clock podf. 00 divide by 1(Default) 01 divide by 2 10 divide by 4 11 divide by 8 Note: Divider should be updated when output clock is gated.
13–11 esdhc1_clk_podf[2:0]	Divider for esdhc1 clock podf. 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
10–8 usboh3_clk_pred[2:0]	Divider for usboh3 clock pred. 000 Restricted 001 divide by 2 010 divide by 3 011 divide by 4(Default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
7–6 usboh3_clk_podf[1:0]	Divider for usboh3 clock podf. 00 divide by 1(Default) 01 divide by 2 10 divide by 3 11 divide by 4 Note: Divider should be updated when output clock is gated.

Table 7-15. CSCDR1 Field Descriptions (continued)

Field	Description
5–3 uart_clk_pred[2:0]	Divider for uart clock pred. 000 divide by 1 001divide by 2 010 divide by 3 011 divide by 4(Default) 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
2–0 uart_clk_podf[2:0]	Divider for uart clock podf. 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.

NOTE

Any change on the above dividers will have to be done while the module that its clock is affected is not functional and the affected clock is gated. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

7.3.3.11 CCM SSI1 Clock Divider Register(CS1CDR)

Figure 7-13 represents the CCM SSI1 Clock Divider Register (CS1CDR). The CS1CDR register contains bits to control the ssi1 clock generation dividers. Table 7-16 provides its field descriptions.

73FD_4028 (CS1CDR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	ssi_ext1_clk_pred [2:0]			ssi_ext1_clk_podf [5:0]					
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	ssi1_clk_pred [2:0]			ss1_clk_podf [5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1

Figure 7-13. CCM SSI1 Clock Divider Register(CS1CDR)

Table 7-16. CS1CDR Field Descriptions

Field	Description
31-25	Reserved
24–22 ssi_ext1_clk_pred[2:0]	Divider for ssi_ext1 clock pred. 000 restricted 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
21–16 ssi_ext1_clk_podf [5:0]	Divider for ssi_ext1 clock podf. 000000divide by 1 111111divide by 2^6 The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
15-9	Reserved

Table 7-16. CS1CDR Field Descriptions (continued)

Field	Description
8–6 ssi1_clk_pred[2:0]	Divider for ssi1 clock pred. 000 restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
5–0 ssi1_clk_podf [5:0]	Divider for ssi1 clock podf. 000000divide by 1 111111divide by 2^6 The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

7.3.3.12 CCM SSI2 Clock Divider Register(CS2CDR)

Figure 7-14 represents the CCM SSI2 Clock Divider Register (CS2CDR). The CS2CDR register contains bits to control the ssi2 clock generation dividers. Table 7-17 provides its field descriptions.

73FD_402c (CS2CDR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	ssi_ext2_clk_pred [2:0]			ssi_ext2_clk_podf [5:0]					
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	ssi2_clk_pred [2:0]			ss2_clk_podf [5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1

Figure 7-14. CCM SSI2 Clock Divider Register(CS2CDR)

Table 7-17. CS2CDR Field Descriptions

Field	Description
31-25	Reserved
24–22 ssi_ext2_clk_pred[2:0]	Divider for ssi_ext2 clock pred. 000 Restricted 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
21–16 ssi_ext2_clk_podf [5:0]	Divider for ssi_ext2 clock podf. 000000divide by 1 111111divide by 2^6 The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
15-9	Reserved
8–6 ssi2_clk_pred[2:0]	Divider for ssi2 clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
5–0 ssi2_clk_podf [5:0]	Divider for ssi2 clock podf. 000000divide by 1 111111divide by 2^6 The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

7.3.3.13 CCM DI Clock Divider Register(CDCDR)

Figure 7-15 represents the CCM DI Clock Divider Register (CDCDR). The CDCDR register contains bits to control the DI clock, tve clock and usb_phy generation dividers. Table 7-18 provides its field descriptions.

73FD_4030 (CDCDR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	tve_clk_pred [2:0]			spdif0_clk_pred [2:0]			spdif0_clk_podf [5:0]						spdif1_clk_pred [2:0]		
W																
Reset	0	0	0	0	0	1	0	0	0	0	1	1	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	spdif1_clk_podf [5:0]						di_clk_pred [2:0]			usb_phy_pred[2:0]			usb_phy_podf[2:0]		
W																
Reset	0	0	0	0	1	1	0	1	1	1	0	1	0	0	1	0

Figure 7-15. CCM DI Clock Divider Register(CDCDR)

Table 7-18. CDCDR Field Descriptions

Field	Description
31-1	Reserved
30–28 tve_clk_pred[2:0]	Divider for tve clock pred. 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
27- 25 spdif0_clk_pred[2:0]	Divider for spdif0 clock pred. 000 restricted 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.

Table 7-18. CDCDR Field Descriptions (continued)

Field	Description
24–19 spdif0_clk_podf [5:0]	Divider for spdif0 clock podf. 000000divide by 1 111111divide by 2^6 Note: Divider should be updated when output clock is gated. Note: The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
18–16 spdif1_clk_pred[2:0]	Divider for spdif1 clock pred. 000 restricted 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
15	Reserved
14–9 spdif1_clk_podf [5:0]	Divider for spdif1 clock podf. 000000divide by 1 111111divide by 2^6 Note: Divider should be updated when output clock is gated. Note: The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
8–6 di_clk_pred[2:0]	Divider for di clock pred. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 (Default) Note: Divider should be updated when output clock is gated.

Table 7-18. CDCDR Field Descriptions (continued)

Field	Description
5–3 usb_phy_pred[2:0]	Divider for usb_phy clock pred. 000 Restricted 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
2–0 usb_phy_podf[2:0]	Divider for usb_phy clock podf. 000 divide by 1 001 divide by 2 010 divide by 3(Default) 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.

7.3.3.14 CCM Serial Clock Divider Register 2(CSCDR2)

Figure 7-16 represents the CCM Serial Clock Divider Register 2(CSCDR2). The CSCDR2 register contains bits to control the clock generation sub module dividers. Table 7-19 provides its field descriptions.

73FD_4038 (CSCDR2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	ecspi_clk_pred[2:0]			ecspi_clk_podf[5:0]					sim_clk_pred[2:0]			
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	sim_clk_podf[5:0]						0	0	1	0	0	0	0	0	1
W																
Reset	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1

Figure 7-16. CCM Serial Clock Divider Register 2(CSCDR2)

Table 7-19. CSCDR2 Field Descriptions

Field	Description
31-28	Reserved
27 -25 ecspi_clk_pred[2:0]	Divider for ecspi clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Divider should be updated when output clock is gated.
24–19 ecspi_clk_podf [5:0]	Divider for ecspi clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
18 -16 sim_clk_pred[2:0]	Divider for sim clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Divider should be updated when output clock is gated.
15	Reserved
14–9 sim_clk_podf [5:0]	Divider for sim clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
8-0	Reserved.

7.3.3.15 CCM Serial Clock Divider Register 3(CSCDR3)

Figure 7-16 represents the CCM Serial Clock Divider Register 3(CSCDR3). The CSCDR3 register contains bits to control the clock generation sub module dividers. Table 7-19 provides its field descriptions.

73FD_403c (CSCDR3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	**0**	**0**	0	**0**	**0**	**0**	**0**	**0**	hsi2c_clk_pred[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	hsi2c_clk_podf[5:0]							firi_clk_pred[2:0]				firi_clk_podf[5:0]			
W																
Reset	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1

Figure 7-17. CCM Serial Clock Divider Register 3(CSCDR3)

Table 7-20. CSCDR3 Field Descriptions

Field	Description
31-28	Reserved
18 -16 hsi2c_clk_pred[2:0]	Divider for sim clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Divider should be updated when output clock is gated.
15	Reserved
14–9 hsi2c_clk_podf [5:0]	Divider for sim clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

Table 7-20. CSCDR3 Field Descriptions (continued)

Field	Description
8–6 firi_clk_pred[2:0]	Divider for firi clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Divider should be updated when output clock is gated.
5–0 firi_clk_podf [5:0]	Divider for firi clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

7.3.3.16 CCM Serial Clock Divider Register 4(CSCDR4)

Figure 7-16 represents the CCM Serial Clock Divider Register 4 (CSCDR4). The CSCDR4 register contains bits to control the clock generation sub module dividers. Table 7-19 provides its field descriptions.

73FD_4040 (CSCDR4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	**0**	**0**	0	**0**	**0**	**0**	**0**	**0**	csi_mclk2_clk_pred[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	csi_mclk2_clk_podf[5:0]							csi_mclk1_clk_pred[2:0]		csi_mclk1_clk_podf[5:0]					
W																
Reset	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	1

Figure 7-18. CCM Serial Clock Divider Register 4(CSCDR4)

Table 7-21. CSCDR4 Field Descriptions

Field	Description
31-28	Reserved
18 -16 csi_mclk2_clk_pred[2:0]	Divider for csi mclk2 clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 **Note:** Divider should be updated when output clock is gated.
15	Reserved
14–9 csi_mclk2_clk_podf [5:0]	Divider for csi mclk2 clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.
8–6 csi_mclk1_clk_pred[2:0]	Divider for csi mclk1 clock pred. 000 Restricted 001 divide by 2(Default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 Note: Divider should be updated when output clock is gated.
5–0 csi_mclk1_clk_podf [5:0]	Divider for csi mclk1 clock podf. 000000divide by 1 111111divide by 2^6 **Note:** Divider should be updated when output clock is gated. **Note:** The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

NOTE

Any change on the serial clock dividers (all dividers in the registers CSCDR1, CECDR, CDCDR, CSCDR2) will have to be done while the module that its clock is affected is not functional. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

7.3.3.17 CCM Wakeup Detector Register(CWDR)

Figure 7-19 represents the CCM Wakeup Detector Register (CWDR). The CWDR register contains bits to control the functionality of the wakeup detector.Table 7-22 provides its field descriptions.

73FD_4044 (CWDR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	gpio1_9_dir	gpio1_8_dir	gpio1_7_dir	gpio1_6_dir	gpio1_5_dir	gpio1_4_dir
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	gpio1_9_icr		gpio1_8_icr		gpio1_7_icr		gpio1_6_icr		gpio1_5_icr		gpio1_4_icr	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-19. CCM Register (CWDR)

Table 7-22. CWDR Field Descriptions

Field	Description
31–22	Reserved
21 gpio1_9_dir	Defines direction of gpio1_9. Software should program this value correspondingly to the settings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
20 gpio1_8_dir	Defines direction of gpio 1_8. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
19 gpio1_7_dir	Defines direction of gpio 1_7. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
18 gpio1_6_dir	Defines direction of gpio 1_6. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
17 gpio1_5_dir	Defines direction of gpio 1_5. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
16 gpio1_4_dir	Defines direction of gpio 1_4. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 0 gpio configures as input 1 gpio configured as output
15–12	Reserved

Table 7-22. CWDR Field Descriptions (continued)

Field	Description
11–10 gpio1_9_ icr	Interrupt sensitivity configuration bits for gpio1_9. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive
9–8 gpio1_8_ icr	Interrupt sensitivity configuration bits for gpio1_8. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive
7–6 gpio1_7_ icr	Interrupt sensitivity configuration bits for gpio1_7. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive
5–4 gpio1_6_ icr	Interrupt sensitivity configuration bits for gpio1_6. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive
3–2 gpio1_5_ icr	Interrupt sensitivity configuration bits for gpio1_5. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive
1–0 gpio1_4_ icr	Interrupt sensitivity configuration bits for gpio1_4. Software should program this value correspondingly to the setings in the GPIO, so that the wakeup detector will be able to track the correct signals. 00 - The interrupt is low level sensitive 01 - The interrupt is high level sensitive 10 - The interrupt is rise edge sensitive 11 - The interrupt is fall edge sensitive

7.3.3.18 CCM Divider Handshake In-Process Register(CDHIPR)

Figure 7-20 represents the CCM Divider Handshake In-Process Register (CDHIPR). The CDHIPR register contains read-only bits that indicate that ccm is in process of updating dividers or muxes that might need handshake with modules.

Bit 16 corresponds to the arm_podf divider only if ARM DVFS operation is done through arm_podf change (arm_freq_shift_divider = ‘1’). In this case, ARM divider will be updated only after assertion of arm_clk_switch_req from GPC. During this period, bit 16 (arm_podf_busy) will assert to indicate that arm_podf is during process of change. Software should not write new values to arm_podf during this period. any reads of arm_podf during this period will result the next value of arm_podf and not the actual

dividers value. To read the actual dividers value, software should wait until arm_podf_busy deasserts. Once the value of the indication bit changes from ‘1’ to ‘0’, ccm can also generate interrupt, if its not masked (refer to CIMR). This bit will not assert if its not a ARM DVFS operation, for example, if arm_freq_shift_divider = ‘0’. In this case, ARM divider will be updated once arm_podf register will be written.

The dividers in the bits 8–0 group are axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, ddr_clk_podf and nfc_podf. The muxes control in this group are ddr_high_freq_clk_sel, periph_clk_sel and emi_clk_sel.

For each of those dividers and muxes control, CDHIPR holds a busy indication bit. If this bit is equal to 1, then ccm is in process of updating the divider or mux. The corresponding bit will assert to ‘1’ once the CBCDR register is updated and a handshake is indeed needed for the update. The corresponding bit will deassert to ‘0’ once the handshake has completed and the divider or mux is loaded with the new values. Software reads of the divider or mux control bits, during the time that these bits are 1, will represent the next value of the divider or mux to be loaded. If software wants to read the actual divider value, it should wait until the value of the indicator bit is equal 0. Once the value of the indication bit changes from ‘1’ to ‘0’, ccm can also generate interrupt, if its not masked (refer to CIMR). Table 7-23 provides its field descriptions.

The handshake bypass should be used in case that a specific module that has a handshake is disabled - in this case the module is not ready to answer for the handshake, and the handshake should be bypassed. In case of a bypass, the CCM will generate the request to the module but will not wait for the acknowledge signal.

73FD_4048 (CDHIPR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	arm_podf_busy
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	ddr_high_freq_clk_sel_busy	ddr_podf_busy	emi_clk_sel_busy	periph_clk_sel_busy	nfc_podf_busy	ahb_podf_busy	emi_slow_podf_busy	axi_b_podf_busy	axi_a_podf_busy
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-20. CCM Divider Handshake In Process Register(CDHIPR)

Table 7-23. CDHIPRR Field Descriptions

Field	Description
31–10	Reserved
16 arm_podf_busy	Busy indicator for arm_podf. This bit corresponds to the arm_podf divider only if ARM DVFS operation is done through arm_podf change (arm_freq_shift_divider = '1'). 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process. The value read in the divider represents the next value that the divider will hold after the handshake with gpc (after assertion of arm_clk_switch_req and the actual write to the divider).
15–9	Reserved
8 ddr_high_freq_clk_sel_busy	Busy indicator for ddr_high_freq_clk_sel mux control. 0 mux is not busy and its value represents the actual division. 1 mux is busy with handshake process with module. The value read in the register represents the next value that the mux will hold after the handshake.
7 ddr_podf_busy	Busy indicator for ddr_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the regster represents the next value that the divider will hold after the handshake.
6 emi_clk_sel_busy	Busy indicator for emi_clk_sel. 0 mux is not busy and its value represents the actual division. 1 mux is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
5 periph_clk_sel_busy	Busy indicator for periph_clk_sel mux control. 0 mux is not busy and its value represents the actual division. 1 mux is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
4 nfc_podf_busy	Busy indicator for nfc_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
3 ahb_podf_busy	Busy indicator for ahb_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
2 emi_slow_podf_busy	Busy indicator for emi_slow_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
1 axi_b_podf_busy	Busy indicator for axi_b_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.
0 axi_a_podf_busy	Busy indicator for axi_a_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the next value that the divider will hold after the handshake.

7.3.3.19 CCM DVFS Control Register(CDCR)

Figure 7-21 represents the CCM DVFS Control Register (CDCR). The CDCR register contains bits to control the DVFS operation.Table 7-24 provides its field descriptions.

73FD_404c (CDCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	sw_periph_clk_div_req_status	sw_periph_clk_div_req	software_DVFS_en	0	0	arm_freq_shift_divider	periph_clk_DVFS_podf[1:0]	
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 7-21. CCM DVFS Control Register (CDCR)

Table 7-24. CDCR Field Descriptions

Field	Description
31-8	Reserved
7 sw_periph_clk_div_req_status	Status bit defines the operation of DVFS driver in case of software control of DVFS divider." Set this bit to '1' to clear - after a DVFS divider switch request it will be asserted and software should write '1' to clear it and prepare it for the next DVFS divider switch request. 1 DVFS frequency switch operation finished. 0 DVFS frequency switch is not finished, or there is no frequency switch request.
6 sw_periph_clk_div_req	Start DVFS frequency division operation. This bit will affect the DVFS divider only if software_DVFS_en bit was set to '1'. 0 Remove DVFS divider operation - DVFS divider will divide by '1'. 1 enable DVFS divider operation. DVFS divider will divide by the value which is programmed in periph_clk_DVFS_podf bits.
5 software_DVFS_en	Defines if the DVFS operation will commence by software bit or by GPC signal "periph_clk_div_req". 0 DVFS operation is started by GPC signal "periph_clk_div_req" 1 DVFS operation is started by software bit sw_periph_clk_div_req
4	reserved **Note:** Functionality is no longer supported. Bit shouldn't be altered and must remain in its default value.
3	reserved **Note:** Functionality is no longer supported. Bit shouldn't be altered and must remain in its default value.

Table 7-24. CDCR Field Descriptions (continued)

Field	Description
2 arm_freq_shift_divider	Define if next DVFS of ARM domain will be through podf change or pll relock. 0 Next ARM DVFS operation is done through PLL relock. The PLL relock process will start once arm_clk_switch_req is asserted. (in this case any new writes to arm_podf will immediately change the frequency, without waiting for arm_clk_switch_req). 1 Next ARM DVFS operation is done through arm_podf change. ccm will hold updates to arm_podf until signal arm_clk_switch_req is asserted.
1–0 periph_clk_DVFS_podf[1:0]	Divider value for next operation of peripheral DVFS. This defines the value of the dividers affecting main_bus_clk. Please refer to Figure 7-40 and Figure 7-41. This divider will take place only during DVFS operation. Please refer to Section 7.4.7.2, Peripheral Clock Domain Frequency Shift for details. 00 restricted 01 divide by 2(Default) 10 divide by 3 11 divide by 4 Note: IPU will not support DVFS operation of division by 3.

7.3.3.20 CCM Testing Observability Register (CTOR)

Figure 7-22 represents the CCM Testing Observability Register (CTOR). CCM includes three muxes to mux between different critical signals for testing observability. The output of the three muxes is generated on the three output signals obs_output_0, obs_output_1 and obs_output_2. Those three output signals can be generated on the IC pads by configuring the IOMUXC. The CTOR register contains bits to control the data generated for observability on the three output signals above. Table 7-25 provides its field descriptions.

73FD_4050 (CTOR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	obs_en	obs_output_0_sel					obs_output_1_sel				obs_output_2_sel			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-22. CCM Testing Observability Register (CTOR)

Table 7-25. CTOR Field Descriptions

Field	Description
31–14	Reserved
13 obs_en	observability enable bit. this bit enables the output of the three observability muxes. 0 Observability mux disabled. 1 Observability mux enabled.
12–8 obs_spare_output_0_sel	Selection of the signal to be generated on obs_output_0 (output of CCM) for observability on the pads. 00000 ccm_system_in_stop_mode 00001lpm_current_state[0] 00010 hndsk_current_state[0] 00011shd_current_state[0] 00100 ccm_ipg_stop 00101ccm_pdn_4arm_req 00110emi_freq_change_req 00111ipu_freq_change_req 01000 Reserved 01001Reserved 01010 pll_lrf_sticky1 01011fpm_lrf 01100clk_src_on 01101ipu_lpsr_wakeup_ack 01110src_warm_DVFS_req 01111periph_clk_div_req 10000arm_clk_switch_req 10001ccm_clk_switch_ack 10010 ipu_clk_changed 10011emi_lpmd 10100emi_lpmd_fast 10101emi_lpmd_int1 10110emi_lpmd_slow 10111 Reserved 11000 11001 11010 obs_input_0 11011 obs_input_1 11100 obs_input_2 11101 obs_input_3 11110 obs_input_4 11111 obs_input_5

Table 7-25. CTOR Field Descriptions (continued)

Field	Description
7–4 obs_spare_output_1_sel	Selection of the signal to be generated on obs_output_1 (output of CCM) for observability on the pads. 0000 ccm_system_in_wait_mode 0001 lpm_current_state[1] 0010 hndsk_current_state[1] 0011 ccm_fpm_en 0100 ccm_ipg_wait 0101 ccm_CAMP_dis 0110 dpll_en_dpllip 0111 ccm_pdn_4all_req 1000 emi_freq_change_ack 1001 ipu_freq_change_ack 1010 Reserved 1011Reserved 1100 arm_dsm_request 1101 ccm_lpsr_ipu 1110 gpc_pup_ack 1111 pll_lrf_sticky2
3–0 obs_spare_output_2_sel	Selection of the signal to be generated on obs_output_2 (output of CCM) for observability on the pads. 0000 ccm_int_mem_ipg_stop 0001 lpm_current_state[2] 0010 hndsk_current_state[2] 0011 shd_current_state[1] 0100 pll_lvs 0101 src_clock_ready 0110 ref_clk_en_dpllip 0111 ccm_pup_req 1000 emi_lpack 1001 emi_lpack_fast 1010 emi_lpack_slow 1011 emi_lpack_int1 1100 src_power_gating_reset_done 1101 tzic_dsm_wakeup 1110 gpc_pdn_ack 1111 pll_lrf_sticky3

7.3.3.21 CCM Low Power Control Register(CLPCR)

Figure 7-23 represents the CCM Low Power Control Register (CLPCR). The CLPCR register contains bits to control the low power modes operation.Table 7-26 provides its field descriptions.

73FD_4054 (CLPCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	bypass_scc_lpm_hs	bypass_max_lpm_hs	bypass_sdma_lpm_hs	bypass_emi_lpm_hs	bypass_ipu_lpm_hs	bypass_rtic_lpm_hs	bypass_sahara_lpm_hs
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	apm_sdma_clk_gate_en_bit	cosc_pwrdown	stby_count		VSTBY	dis_ref_osc	SBYOS	ARM_clk_dis_on_lpm	lpsr_clk_sel		bypass_pmic_vfunctional_ready	LPM[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1

Figure 7-23. CCM Low Power Control Register (CLPCR)

Table 7-26. CLPCR Field Descriptions

Field	Description
31–24	Reserved
23	Reserved **Note:** Functionality is no longer supported. Bit must be set for better power saving - along with configuration of CCDR[18] and CCGR4[13:6]. Please refer to description of these bits for the recommended values.
22 bypass_scc_lpm_hs	Bypass handshake with scc on next entrance to STOP mode. CCM does not wait for the module's acknowledge. 0 handshake with SCC on next entrance to STOP mode will be performed. (default). 1 handshake with SCC on next entrance to STOP mode will be bypassed.
21 bypass_max_lpm_hs	Bypass handshake with max on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with max on next entrance to low power mode will be performed. (default). 1 handshake with max on next entrance to low power mode will be bypassed.
20 bypass_sdma_lpm_hs	Bypass handshake with sdma on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with sdma on next entrance to low power mode will be performed. (default). 1 handshake with sdma on next entrance to low power mode will be bypassed.

Table 7-26. CLPCR Field Descriptions (continued)

Field	Description
19 bypass_emi_lpm_hs	Bypass handshake with emi on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with emi on next entrance to low power mode will be performed. (default). 1 handshake with emi on next entrance to low power mode will be bypassed.
18 bypass_ipu_lpm_hs	Bypass handshake with IPU on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with IPU on next entrance to low power mode will be performed. (default). 1 handshake with IPU on next entrance to low power mode will be bypassed.
17 bypass_rtic_lpm_hs	Bypass handshake with rtic on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with rtic on next entrance to low power mode will be performed. (default). 1 handshake with rtic on next entrance to low power mode will be bypassed.
16 bypass_sahara_lpm_hs	Bypass handshake with SAHARA on next entrance to low power mode (wait or STOP mode). CCM does not wait for the module's acknowledge. 0 handshake with SAHARA on next entrance to low power mode will be performed. (default). 1 handshake with SAHARA on next entrance to low power mode will be bypassed.
15–13	Reserved
12 apm_sdma_clk_gate_en_bit	This bit will allow the operation of automatic clock gating of SDMA - to be used in LP-APM mode. 1 Enable automatic clock gating of SDMA clocks. For this feature to work, the CGR bits of SDMA should be programmed to turn the clocks on in WAIT mode. 0 Disable automatic clock gating of SDMA clocks. In this case the sdma_clk_enable signal will operate as indicated by the CGR bits.
11 cosc_pwrdown	In run mode, software can manually control powering down of the on-chip oscillator, that is, generating '1' on cosc_pwrdown signal. If software manually powered down the on-chip oscillator, then sbyos functionality for on-chip oscillator will be bypassed. The manual closing of the on-chip oscillator should be performed only in case the reference oscilator is not the source of all the clocks generation. 0 On chip oscillator will not be powered down, that is, cosc_pwrdown = '0'.(default) 1 On chip oscillator will be powered down, that is, cosc_pwrdown = '1'.
9–10 stby_count	Standby counter definition. These two bits define, in the case of STOP exit (if vstby bit was set), the amount of time CCM will wait between PMIC_VSTBY_REQ negation and the check of assertion of PMIC_VFUNCIONAL_READY. If the counter is used when the handshake is bypassed, then add this here. 00 CCM will wait 2 CKIL clock cycles 01 CCM will wait 4 CKIL clock cycles 10 CCM will wait 8 CKIL clock cycles 11 CCM will wait 16 CKIL clock cycles
8 VSTBY	Voltage standby request bit. This bit defines if PMIC_VSTBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage, will be asserted in STOP mode. 0 voltage will not be changed to standby voltage after next entrance to STOP mode. (PMIC_VSTBY_REQ will remain negated - '0') 1 voltage will be requested to change to standby voltage after next entrance to STOP mode. (PMIC_VSTBY_REQ will be asserted - '1'). Note: When returning from STOP mode, the PMIC_VSTBY_REQ will be deasserted (if it was asserted when entering STOP mode), and CCM will wait for indication that functional voltage is ready (by sampling the assertion of pmic_vfuncional_ready) before continuing the process of exiting from STOP mode. Please refer to stby_count bits.

Table 7-26. CLPCR Field Descriptions (continued)

Field	Description
7 dis_ref_osc	dis_ref_osc - in run mode, software can manually control closing of external reference oscillator clock, that is, generating '1' on ref_en_b signal. If software closed manually the external reference clock, then sbyos functionality will be bypassed. The manual closing of external reference oscilator should be performed only in case the reference oscilator is not the source of any clock generation. 0 external high frequency oscillator will be enabled, that is, ref_en_b = '0'.(default) 1 external high frequency oscillator will be disabled, that is, ref_en_b = '1'
6 SBYOS	Standby clock oscillator bit. This bit defines if REF_EN_B pin, which disables external high frequency crystal, and cosc_pwrdown, which power down the on-chip oscillator, will be asserted in STOP mode. This bit is discarded if dis_ref_osc = '1' for external oscillator, and if cosc_pwrdown='1' for the on-chip oscillator. 0 external high frequency oscillator will not be disabled and the on-chip oscillator will not be powered down, after next entrance to STOP mode. (ref_en_b will remain asserted - '0' and cosc_pwrdown will remain de asserted - '0') 1 external high frequency oscillator will be disabled and the on-chip oscillator will be powered down, after next entrance to STOP mode. (ref_en_b will be deasserted - '1' and cosc_pwrdown will be asserted - '1'). (default). When returning from STOP mode, external oscillator will be enabled again, the on-chip oscillator will return to oscillator mode, and after oscnt count ccm will continue with the exit from STOP mode process.
5 ARM_clk_dis_on_lpm	Define if ARM clocks (arm_clk, soc_mxclk, soc_pclk, soc_dbg_pclk, vl_wrck) will be disabled on wait mode. This is useful for debug mode, when the user wants to simulate entering wait mode and keep ARM clock functioning. 0 ARM clock enabled on wait mode. 1 ARM clock disabled on wait mode. (default). Note: software should not enable ARM power gating in wait mode if this bit is cleared.
4–3 lpsr_clk_sel	Controlls the mux to select the lpsr_clk_root generation. Refer to Figure 7-46 and to 7.4.8.4, "Low Power Screen Refresh Mode (LPSR) for details. 00 Reserved 01 generate clock from FPM 10 generate clock from FPM/2 11 GND - to be set when LPSR is not used, (for power consumption considerations) (default).
2 bypass_pmic_vfunctional_ready	By asserting this bit CCM will bypass waiting for pmic_vfunctional_ready signal when coming out of STOP mode. This should be used for PMIC's that don't support the pmic_vfunctional_ready signal. 0 Don't bypass the pmic_vfunctional_ready signal - CCM will wait for it's assertion during exit of low power mode if standby voltage was enabled. 1 bypass the pmic_vfunctional_ready signal - CCM will not wait for it's assertion during exit of low power mode if standby voltage was enabled.
1–0 LPM[1:0]	Setting the low power mode that system will enter on next assertion of dsm_request signal. 00 Remain in run mode 01 Transfer to wait mode 10 Transfer to STOP mode 11 Transfer to LPSR mode (Low Power Screen Refresh Mode).

7.3.3.22 CCM Interrupt Status Register(CISR)

Figure 7-24 represents the CCM Interrupt Status Register (CISR). This is a write one to clear register. Once a interrupt is generated, software should write one to clear it. Table 7-27 provides its field descriptions.

73FD_4058 (CISR) Access: User write one to clear

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	arm_podf_loaded	ddr_high_freq_clk_sel_loaded	ddr_clk_podf_loaded	emi_clk_sel_loaded	periph_clk_sel_loaded	nfc_podf_loaded	ahb_podf_loaded	emi_slow_podf_loaded	axi_b_podf_loaded	axi_a_podf_loaded	dividers_loaded
W						w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	gpio1_9_wakeup_det	gpio1_8_wakeup_det	gpio1_7_wakeup_det	gpio1_6_wakeup_det	gpio1_5_wakeup_det	gpio1_4_wakeup_det	sdhc2_wakeup_det	sdhc1_wakeup_det	kpp_wakeup_det	cosc_ready	CAMP2_ready	CAMP1_ready	fpm_ready	lrf_pll3	lrf_pll2	lrf_pll1
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7-24. CCM Interrupt Status Register (CISR)

Table 7-27. CISR Field Descriptions

Field	Description
31–27	Reserved
26 arm_podf_loaded	Interrupt ipi_int_1 generation. due to frequency change of arm_podf. The interrupt will commence only if arm_podf is loaded during a arm DVFS operation. 0 - interrupt is not genrerated due to frequency change of arm_podf. The frequency of arm_podf is not changed, the values are changed which cause the frequency to change. The same is also relevant for next fields and for description of the interrupt mask register. 1 - interrupt genrerated due to frequency change of arm_podf
25 ddr_high_freq_clk_sel_loaded	Interrupt ipi_int_1 generation due to update of ddr_high_freq_clk_sel. 0 - interrupt is not genrerated due to update of ddr_high_freq_clk_sel. 1 - interrupt genrerated due to update of ddr_high_freq_clk_sel.
24 ddr_clk_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of ddr_podf 0 - interrupt is not genrerated due to frequency change of ddr_podf 1 - interrupt genrerated due to frequency change of ddr_podf
23 emi_clk_sel_loaded	Interrupt ipi_int_1 generation due to update of emi_clk_sel. 0 - interrupt is not genrerated due to update of emi_clk_sel. 1 - interrupt genrerated due to update of emi_clk_sel.

Table 7-27. CISR Field Descriptions (continued)

Field	Description
22 periph_clk_sel_loaded	Interrupt ipi_int_1 generation due to update of periph_clk_sel. 0 - interrupt is not genrerated due to update of periph_clk_sel. 1 - interrupt genrerated due to update of periph_clk_sel.
21 nfc_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of nfc_podf 0 - interrupt is not genrerated due to frequency change of nfc_podf 1 - interrupt genrerated due to frequency change of nfc_podf
20 ahb_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of ahb_podf 0 - interrupt is not genrerated due to frequency change of ahb_podf 1 - interrupt genrerated due to frequency change of ahb_podf
19 emi_slow_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of emi_slow_podf 0 - interrupt is not genrerated due to frequency change of emi_slow_podf 1 - interrupt genrerated due to frequency change of emi_slow_podf
18 axi_b_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of axi_b_podf 0 - interrupt is not genrerated due to frequency change of axi_b_podf 1 - interrupt genrerated due to frequency change of axi_b_podf
17 axi_a_podf_loaded	Interrupt ipi_int_1 generation due to frequency change of axi_a_podf 0 - interrupt is not genrerated due to frequency change of axi_a_podf 1 - interrupt genrerated due to frequency change of axi_a_podf
16 dividers_loaded	Interrupt ipi_int_1 is generated due to frequency update which is cause by programming any of the following: axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, periph_clk_sel, emi_clk_sel. 0 - interrupt is not genrerated due to updated of axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, periph_clk_sel, emi_clk_sel. 1 - interrupt genrerated due to updated of axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, periph_clk_sel, emi_clk_sel.
15 gpio1_9_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_9 input. 0 - interrupt is not genrerated due to change of gpio1_9 input 1 - interrupt genrerated due to change of gpio1_9 input
14 gpio1_8_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_8 input. 0 - interrupt is not genrerated due to change of gpio1_8 input 1 - interrupt genrerated due to change of gpio1_8 input
13 gpio1_7_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_7 input. 0 - interrupt is not genrerated due to change of gpio1_7 input 1 - interrupt genrerated due to change of gpio1_7 input
12 gpio1_6_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_6 input. 0 - interrupt is not genrerated due to change of gpio1_6 input 1 - interrupt genrerated due to change of gpio1_6 input
11 gpio1_5_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_5 input. 0 - interrupt is not genrerated due to change of gpio1_5 input 1 - interrupt genrerated due to change of gpio1_5 input
10 gpio1_4_wakeup_det	Interrupt ipi_int_2 generation due to change of gpio1_4 input. 0 - interrupt is not genrerated due to change of gpio1_4 input 1 - interrupt genrerated due to change of gpio1_4 input

Table 7-27. CISR Field Descriptions (continued)

Field	Description
9 sdhc2_wakeup_det	Interrupt ipi_int_2 generation due to insertion of sdhc2 card. 0 - interrupt is not genrerated due to insertion of sdhc2 card 1 - interrupt genrerated due to insertion of sdhc2 card
8 sdhc1_wakeup_det	Interrupt ipi_int_2 generation due to insertion of sdhc1 card. 0 - interrupt is not genrerated due to insertion of sdhc1 card 1 - interrupt genrerated due to insertion of sdhc1 card
7 kpp_wakeup_det	Interrupt ipi_int_2 generation due to key press detection. 0 - interrupt is not genrerated due to key press detection 1 - interrupt genrerated due to key press detection
6 cosc_ready	Interrupt ipi_int_2 generation due to on-chip oscillator ready, that is, oscnt has finished counting. 0 - interrupt is not genrerated due to on-chip oscillator ready 1 - interrupt genrerated due to on-chip oscillator ready
5 CAMP2_ready	Interrupt ipi_int_2 generation due to CAMP2 ready, that is, oscnt has finished counting. 0 - interrupt is not genrerated due to CAMP2 ready 1 - interrupt genrerated due to CAMP2 ready
4 CAMP1_ready	Interrupt ipi_int_2 generation due to CAMP1 ready, that is, oscnt has finished counting. 0 - interrupt is not genrerated due to CAMP1 ready 1 - interrupt genrerated due to CAMP1 ready
3 fpm_ready	Interrupt ipi_int_2 generation due to fpm ready 0 - interrupt is not genrerated due to FPM ready 1 - interrupt genrerated due to FPM ready
2 lrf_pll3	Interrupt ipi_int_2 generation due to lock of pll3 0 - interrupt is not genrerated due to lock ready of pll_3 1 - interrupt genrerated due to lock ready of pll_3
1 lrf_pll2	Interrupt ipi_int_2 generation due to lock of pll2 0 - interrupt is not genrerated due to lock ready of pll_2 1 - interrupt genrerated due to lock ready of pll_2
0 lrf_pll1	Interrupt ipi_int_2 generation due to lock of pll1 0 - interrupt is not genrerated due to lock ready of pll_1 1 - interrupt genrerated due to lock ready of pll_1

7.3.3.23 CCM Interrupt Mask Register(CIMR)

Figure 7-25 represents the CCM Interrupt Mask Register (CIMR). Table 7-28 provides its field descriptions.

73FD_405c (CIMR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1	1	1	1	1	mask_arm_podf_loaded	mask_ddr_high_freq_clk_sel_loaded	mask_ddr_podf_loaded	mask_emi_clk_sel_loaded	mask_periph_clk_sel_loaded	mask_nfc_podf_loaded	mask_ahb_podf_loaded	mask_emi_slow_podf_loaded	mask_axi_b_podf_loaded	mask_axi_a_podf_loaded	mask_dividers_loaded
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	mask_gpio1_9_wakeup_det	mask_gpio1_8_wakeup_det	mask_gpio1_7_wakeup_det	mask_gpio1_6_wakeup_det	mask_gpio1_5_wakeup_det	mask_gpio1_4_wakeup_det	mask_sdhc2_wakeup_det	mask_sdhc1_wakeup_det	mask_kpp_wakeup_det	mask_cosc_ready	mask_CAMP2_ready	mask_CAMP1_ready	mask_fpm_ready	mask_lrf_pll3	mask_lrf_pll2	mask_lrf_pll1
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-25. CCM Interrupt Mask Register (CIMR)

Table 7-28. CIMR Field Descriptions

Field	Description
31–27	Reserved
26 arm_podf_loaded	Mask interrupt generation due to frequency change of arm_podf 0 - don't mask interrupt due to frequency change of arm_podf - interrupt will be created 1 - mask interrupt due to frequency change of mask_arm_podf_loaded"
25 mask_ddr_high_freq_clk_sel_loaded	mask interrupt generation due to update of ddr_high_freq_clk_sel. 0 - don't mask interrupt due to update of ddr_high_freq_clk_sel - interrupt will be created 1 - mask interrupt due to update of ddr_high_freq_clk_sel
24 mask_ddr_podf_loaded	mask interrupt generation due to frequency change of ddr_podf 0 - don't mask interrupt due to frequency change of ddr_podf - interrupt will be created 1 - mask interrupt due to frequency change of ddr_podf
23 mask_emi_clk_sel_loaded	mask interrupt generation due to update of emi_clk_sel. 0 - don't mask interrupt due to update of emi_clk_sel - interrupt will be created 1 - mask interrupt due to update of emi_clk_sel
22 mask_periph_clk_sel_loaded	mask interrupt generation due to update of periph_clk_sel. 0 - don't mask interrupt due to update of periph_clk_sel - interrupt will be created 1 - mask interrupt due to update of periph_clk_sel

Table 7-28. CIMR Field Descriptions (continued)

Field	Description
21 mask_nfc_podf_loaded	mask interrupt generation due to frequency change of nfc_podf 0 - don't mask interrupt due to frequency change of nfc_podf - interrupt will be created 1 - mask interrupt due to frequency change of nfc_podf
20 mask_ahb_podf_loaded	mask interrupt generation due to frequency change of ahb_podf 0 - don't mask interrupt due to frequency change of ahb_podf - interrupt will be created 1 - mask interrupt due to frequency change of ahb_podf
19 mask_emi_slow_podf_loaded	mask interrupt generation due to frequency change of emi_slow_podf 0 - don't mask interrupt due to frequency change of emi_slow_podf - interrupt will be created 1 - mask interrupt due to frequency change of emi_slow_podf
18 mask_axi_b_podf_loaded	mask interrupt generation due to frequency change of axi_b_podf 0 - don't mask interrupt due to frequency change of axi_b_podf - interrupt will be created 1 - mask interrupt due to frequency change of axi_b_podf
17 mask_axi_a_podf_loaded	mask interrupt generation due to frequency change of axi_a_podf 0 - don't mask interrupt due to frequency change of axi_a_podf - interrupt will be created 1 - mask interrupt due to frequency change of axi_a_podf
16 mask_dividers_loaded	mask interrupt generation due to updated of axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, ddr_podf, ddr_high_freq_clk_sel, periph_clk_sel, emi_clk_sel. 0 - don't mask interrupt due to updated of axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, ddr_podf, ddr_high_freq_clk_sel, periph_clk_sel, emi_clk_sel. 1 - mask interrupt due to updated of axi_a_podf, axi_b_podf, emi_slow_podf, ahb_podf, nfc_podf, ddr_podf, ddr_high_freq_clk_sel, periph_clk_sel, emi_clk_sel.
15 mask_gpio1_9_wakeup_det	Mask Interrupt generation due to change of gpio1_9 input. 0 - don't mask interrupt due to change of gpio1_9 input 1 - mask interrupt due to change of gpio1_9 input
14 mask_gpio1_8_wakeup_det	Mask Interrupt generation due to change of gpio1_8 input. 0 -don't mask interrupt due to change of gpio1_8 input 1 - mask interrupt due to change of gpio1_8 input
13 mask_gpio1_7_wakeup_det	Mask Interrupt generation due to change of gpio1_7 input. 0 - don't mask interrupt due to change of gpio1_7 input 1 -mask interrupt due to change of gpio1_7 input
12 mask_gpio1_6_wakeup_det	Mask Interrupt generation due to change of gpio1_6 input. 0 - don't mask interrupt due to change of gpio1_6 input 1 - mask interrupt due to change of gpio1_6 input
11 mask_gpio1_5_wakeup_det	Mask Interrupt generation due to change of gpio1_5 input. 0 - don't mask interrupt due to change of gpio1_5 input 1 - mask interrupt due to change of gpio1_5 input
10 mask_gpio1_4_wakeup_det	Mask Interrupt generation due to change of gpio1_4 input. 0 - don't mask interrupt due to change of gpio1_4 input 1 - mask interrupt due to change of gpio1_4 input
9 mask_sdhc2_wakeup_det	Mask Interrupt generation due to insertion of sdhc2 card. 0 - don't mask interrupt due to insertion of sdhc2 card 1 - mask interrupt due to insertion of sdhc2 card
8 mask_sdhc1_wakeup_det	Mask Interrupt generation due to insertion of sdhc1 card. 0 - don't mask interrupt due to insertion of sdhc1 card 1 - mask interrupt due to insertion of sdhc1 card

Table 7-28. CIMR Field Descriptions (continued)

Field	Description
7 mask_kpp_wakeup_det	Mask interrupt generation due to key press detection. 0 - don't mask interrupt due to key press detection 1 - mask interrupt due to key press detection
6 mask_cosc_ready	mask interrupt generation due to on-chip oscillator ready 0 - don't mask interrupt due to on-chip oscillator ready - interrupt will be created 1 - mask interrupt due to on-chip oscillator ready
5 mask_CAMP2_ready	mask interrupt generation due to CAMP2 ready 0 - don't mask interrupt due to CAMP2 ready - interrupt will be created 1 - mask interrupt due to CAMP2 ready
4 mask_CAMP1_ready	mask interrupt generation due to CAMP1 ready 0 - don't mask interrupt due to CAMP1 ready - interrupt will be created 1 - mask interrupt due to CAMP1 ready
3 mask_fpm_ready	mask interrupt generation due to FPM ready 0 - don't mask interrupt due to FPM ready - interrupt will be created 1 - mask interrupt due to FPM ready
2 mask_lrf_pll3	mask interrupt generation due to lrf of pll3 0 - don't mask interrupt due to lrf of pll3 - interrupt will be created 1 - mask interrupt due to lrf of pll3
1 mask_lrf_pll2	mask interrupt generation due to lrf of pll2 0 - don't mask interrupt due to lrf of pll2 - interrupt will be created 1 - mask interrupt due to lrf of pll2
0 mask_lrf_pll1	mask interrupt generation due to lrf of pll1 0 - don't mask interrupt due to lrf of pll1 - interrupt will be created 1 - mask interrupt due to lrf of pll1

7.3.3.24 CCM Clock Output Source Register (CCOSR)

Figure 7-26 represents the CCM Clock Output Source Register (CCOSR). The CCOSR register contains bits to control the clocks that will be generated on the output ipp_do_clko1 and ipp_do_clko2 (also named as CLKO and CLKO2 signals that can selected on i.MX51 pins through IOMUX).Table 7-29 provides its field descriptions.

73FD_4060 (CCOSR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	cko2_en	cko2_div[2:0}				cko2_sel[4:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	cko1_en	cko1_div[2:0}			cko1_sel[3:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 7-26. CCM Clock Output Source Register (CCOSR)

Table 7-29. CCOSR Field Descriptions

Field	Description
31–25	Reserved
24 cko2_en	Enable of CKO2 clock 0 CKO2 disabled. 1 CKO2 enabled.
23–21 cko2_div[2:0]	Setting the divider of CKO2 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8

Table 7-29. CCOSR Field Descriptions (continued)

Field	Description
20–16 cko2_sel[4:0]	Selection of the clock to be generated on cko2 00000dptc_core (this is the ccm input dptc_core1_clk_clko generated to the clko2 BGA contact) 00001 dptc_periph (this is the ccm input dptc_peripheral1_clk_clko generated to the clko2 BGA contact). 00010 Reserved 00011esdhc1_mshc1_clk_root 00100usboh3_clk_root 00101 wrck_clk _root 00110cspi_clk_root 00111pll1_ref_clk 01000 esdhc3_clk_root 01001ddr_clk_root 01010arm_axi_clk_root (Default) 01011 usbphy_pll_out_480 01100vpu_rclk_root 01101ipu_hsp_clk_root 01110 osc_clk 01111 ckih_CAMP1_clk 10000 fpm_clk 10001 esdhc2_clk_root 10010ssi1_clk_root 10011ssi2_clk_root 10100 Reserved 10101 reserved 10110lpsr_clk_root 10111 pgc_clk_root 11000 tve_ext_clk 11001usb_phy_clk_root 11010 tve_216_54_clk_root 11011lp_apm clock 11100 uart_clk_root 11101spdif0_clk_root 11110 spdif1_clk_root 11111spare_input_1 div 8 (connected in SOC to async ref clock for reference clock generation of ARM) This clock source is always divided by 8 before supplying this clock to the cko2 mux.
15–8	Reserved
7 cko1_en	Enable of CKO1 clock 0 CKO1 disabled. 1 CKO1 enabled.

Table 7-29. CCOSR Field Descriptions (continued)

Field	Description
6–4 cko1_div[2:0]	Setting the divider of CKO1 000 divide by 1(Default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
3–0 cko1_sel[3:0]	Selection of the clock to be generated on cko1 0000 arm_clk_root 0001 pll1_sw_clk (Default) 0010 pll2_sw_clk 0011 pll3_sw_clk 0100 emi_core_clk_root 0101 Reserved 0110 enfc_clk_root 0111 Reserved 1000 di_clk_root 1001 Reserved 1010 Reserved 1011 ahb_clk_root 1100 ipg_clk_root 1101 perclk_root 1110 ckil_sync_clk_root 1111 reserved

7.3.3.25 CCM General Purpose Register(CGPR)

Figure 7-27 represents the CCM General Purpose Register (CGPR). This is a regular read/write implemented register, which can serve for future possible usage. the respective bits are connected to cgpr_dout[31:0] output of ccm. Few bits are already in use, The other bits are free to be used in future usage. Table 7-30 provides its field descriptions.

73FD_4064 (CGPR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	0	0	0	0	0	0	0	0	arm_async_ref_en	arm_async_ref_sel						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	arm_async_ref_sel[5]	1	1	1	1	1	1	0	0	1	overide_apm_emi_int1_clock_gating	efuse_prog_supply_gate	FPM_mux_select	0	1	0
Reset	1	1	1	1	1	1	1	0	0	1	1	0	0	0	1	0

Figure 7-27. CCM Register (CGPR)

Table 7-30. CGPR Field Descriptions

Field	Description
31– 24	Reserved for future use. Those bits are connected to ccm output cgpr_dout[31-10]
23 arm_async_ref_en	Enable of ARM async ref circuit. This bit is connected to ccm output cgpr_dout[23]. 0 - ARM async ref circuit disabled 1- ARM async ref circuit enabled
22 arm_async_ref_sel[7]	Bit 7 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 7 (this is in case all other arm_async_ref_sel are defined as ‘0’.
21 arm_async_ref_sel[6]	Bit 6 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 6 (this is in case all other arm_async_ref_sel are defined as ‘0’.
20 arm_async_ref_sel[4]	Bit 4 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 4 (this is in case all other arm_async_ref_sel are defined as ‘0’.

Table 7-30. CGPR Field Descriptions (continued)

Field	Description
19 arm_async_ref_sel[3]	Bit 3 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 3 (this is in case all other arm_async_ref_sel are defined as ‘0’.
18 arm_async_ref_sel[2]	Bit 2 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 2 (this is in case all other arm_async_ref_sel are defined as ‘0’.
17 arm_async_ref_sel[1]	Bit 1of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 1 (this is in case all other arm_async_ref_sel are defined as ‘0’.
16 arm_async_ref_sel[0]	Bit 0 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 0 (this is in case all other arm_async_ref_sel are defined as ‘0’.
15 arm_async_ref_sel[5]	Bit 5 of frequency select of async reference circuit. 0 frequency of async reference circuit is defined by other arm_async_ref_sel bits 1 frequency of async reference circuit is reference number 5 (this is in case all other arm_async_ref_sel are defined as ‘0’.
14–6	Reserved for future use. Those bits are connected to ccm output cgpr_dout[14-6]
5 overide_apm_emi_intr_clock_gating	Defines override of the automatic clock gating control of EMI clock: aclk_intr. If this bit is defined as ‘0’, there will not be an override, and the aclk_intr will be gated each time both m3_clk_gate_en and m4_clk_gate_en are ‘0’. If this bit is ‘1’, then aclk_intr will not be gated based on the above two signals. In APM, this feature will aid consuming less power since EMI aclk_intr will be gated each time there is no access of ARM or SDMA. 1 override aclk_intr input of EMI. 0 don’t override the aclk_intr automatic control, that is, allow control of m3_clk_gate_en and m4_clk_gate_en on aclk_intr.
4 efuse_prog_supply_gate	Defines the value of the output signal cgpr_dout[4]. Gate of programming supply for efuse programing 0 fuse programing supply voltage is gated off to the efuse module 1 allow fuse programing.
3 FPM_mux_select	FPM_mux_select - Defines the value of the output signal cgpr_dout[3] going to FPM. FPM uses this signal to mux between FPM output and FPM internal oscilator. This functionality is used for testability of FPM. Refer to FPM spec for further details. 0 fpm_mux_select output is 0 1 fpm_mux_select output is 1
2–0	Reserved

NOTE

ARM_CLK_ROOT is muxed in top level between the ccm_arm_clock_root output and the clock generated by reference circuit. The frequency of the reference circuit is defined by 8 control frequency select bits of the async reference circuit. Only one single select bit should equal to '1' - all the other select bits should equal to '0'. The frequency is defined by the next table:

Table 7-31. Async Reference Circuit Settings

Field	Description
arm_async_ref_sel[7:0] = 00000001	Frequency Option 0
arm_async_ref_sel[7:0] = 00000010	Frequency Option 1
arm_async_ref_sel[7:0] = 00000100	Frequency Option 2
arm_async_ref_sel[7:0] = 00001000	Frequency Option 3
arm_async_ref_sel[7:0] = 00010000	Frequency Option 4
arm_async_ref_sel[7:0] = 00100000	Frequency Option 5
arm_async_ref_sel[7:0] = 01000000	Frequency Option 6
arm_async_ref_sel[7:0] = 10000000	Frequency Option 7

7.3.3.26 CCM Clock Gating Register(CCGR)

Figure 7-28 through Figure 7-33 represents the CCM Clock Gating Register (CCGR). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 7 CGR registers. The number of registers required is according to the number of peripherals in the system. Table 7-32 through Table 7-38 provides its field descriptions.

73FD_4068 (CCGR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-28. CCM Clock Gating Register (CCGR0)

73FD_406c (CCGR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-29. CCM Clock Gating Register (CCGR1)

73FD_4070 (CCGR2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-30. CCM Clock Gating Register (CCGR2)

73FD_4074 (CCGR3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-31. CCM Clock Gating Register (CCGR3)

73FD_4078 (CCGR4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-32. CCM Clock Gating Register (CCGR4)

73FD_407c (CCGR5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-33. CCM Clock Gating Register (CCGR5)

73FD_4080 (CCGR6) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-34. CCM Clock Gating Register (CCGR6)

7.3.4 CG(i) bits

This bits are used to turn on/off the clock to each module independently. Table 7-32 details the possible clock activity conditions for each module.

Table 7-32. CG Bit Description

CGR value	Clock Activity Description
00	Clock is off during all modes. stop enter hardware handshake is disabled.
01	Clock is on in run mode, but off in WAIT and STOP modes
10	Not applicable (Reserved).
11	clock is on during all modes, except STOP mode.

Note: Stop the module before clearing its bits because clocks to the module will be stopped immediately.

The following tables show the register mappings for the different CGRs. The clock connectivity table should be used to match the “CCM output affected” to the actual clocks going into the modules.

Table 7-33. CCGR0 Register Mapping

CG bit index	Controlled Clocks	Comments	CCM output affected
0	arm_bus	to control ipg bus of arm	arm_bus_clk_enable
1	arm_axi	to control arm_axi input clock	arm_axi_clk_enable
2	arm_debug	to control pclk and atclk busses of arm	arm_debug_clk_enable
3	tzic	to control tzic clocks	tzic_clk_enable
4	dap clocks		dap_clk_enable
5	tpiu clocks		tpiu_clk_enable
6	cti2 clocks		cti2_clk_enable
7	cti3 clocks		cti3_clk_enable
8	ahbmux1 clocks		ahbmux1_clk_enable
9	ahbmux2 clocks		ahbmux2_clk_enable
10	romcp clocks		romcp_clk_enable
11	rom clocks		rom_clk_enable
12	aips_tz1 clocks		aips_tz1_clk_enable
13	aips_tz2 clocks		aips_tz2_clk_enable
14	ahb_max clocks		ahb_max_clk_enable
15	iim clocks	affects also fusebox_poly_128_1, fusebox_poly_256_1, fusebox_poly_256_2, fusebox_poly_256_3	iim_clk_enable

Table 7-34. CCGR1 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	tmax1 clocks		tmax1_clk_enable
1	tmax2 clocks		tmax2_clk_enable
2	tmax3 clocks		tmax3_clk_enable
3	uart1_ipg_clk	affects ipg_clk input to uart1	uart1_clk_enable
4	uart1_perclk	affects ipg_perclk input to uart1	uart1_serial_clk_enable
5	uart2_ipg_clk	affects ipg_clk input to uart2	uart2_clk_enable
6	uart2_perclk	affects ipg_perclk input to uart2	uart2_serial_clk_enable
7	uart3_ipg_clk	affects ipg_clk input to uart3	uart3_clk_enable
8	uart3_perclk	affects ipg_perclk input to uart3	uart3_serial_clk_enable
9	i2c1 clocks		i2c1_serial_clk_enable
10	i2c2 clocks		i2c2_serial_clk_enable
11	hsi2c_ipg_clk	affects hsi2c ipg clk	hsi2c_clk_enable
12	hs2ic_serial clock	affects hsi2c ipg_hsi2c_clk (the serial clock)	hsi2c_serial_clk_enable
13	firi ipg clock		firi_clk_enable
14	firi serial clock		firi_serial_clk_enable
15	SCC clocks	affects scc_clk_enable. it is not possible to close SCC clocks during run mode. writting ‘00’ to this cgr will not affect the clocks of SCC. Closing the clocks of SCC will eventually block the access to the internal memory. In WAIT mode the SCC clocks can be closed only if it is guaranteed that no master will access the internal memory during the time spend in WAIT mode. The SCC clocks will be closed in WAIT mode only if SAHARA clocks were closed as well. During WAIT mode, if SCC will assert either hclk_en signal or ipg_clk_en signal, the CCM will assert scc_clk_enable and will thus enable the SCC clocks. The WAIT mode will not be exited due to this assertion.	scc_clk_enable

Table 7-35. CCGR2 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	usb phy clock		usb_phy_clk_enable
1	epit1_ipg_clk	affects ipg_clk input of epit1	epit1_clk_enable
2	epit1_highfreq	affects ipg_clk_highfreq input of epit1	epit1_serial_clk_enable
3	epit2_ipg_clk	affects ipg_clk input of epit2	epit2_clk_enable

MCG bit index	Controlled Clocks	Comments	CCM output affected
4	epit2_highfreq	affects ipg_clk_highfreq input of epit2	epit2_serial_clk_enable
5	pwm1_ipg_clk	affects ipg_clk input of pwm1	pwm1_clk_enable
6	pwm1_highfreq	affects ipg_clk_highfreq input of pwm1	pwm1_serial_clk_enable
7	pwm2_ipg_clk	affects ipg_clk input of pwm2	pwm2_clk_enable
8	pwm2_highfreq	affects ipg_clk_highfreq input of pwm2	pwm2_serial_clk_enable
9	gpt_ipg_clk	affects ipg_clk input of gpt	gpt_clk_enable
10	gpt_highfreq	affects ipg_clk_highfreq input of gpt	gpt_serial_clk_enable
11	owire clocks		owire_serial_clk_enable
12	fec clocks	affects also csync_fec	fec_clk_enable
13	usboh3_ipg_ahb	affects ipg_clk and ipg_ahb_clk inputs of usboh3	usboh3_clk_enable
14	usboh3_60M	affects ipg_clk_60Mhz input of usboh3	usboh3_serial_clk_enable
15	tve clock		tve_clk_enable

Table 7-36. CCGR3 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	esdhc1_ipg_hclk	affects ipg_clk and hclk inputs of esdhc1	esdhc1_clk_enable
1	esdhc1_perclk	affects ipg_clk_perclk input of esdhc1	esdhc1_serial_clk_enable
2	esdhc2_ipg_hclk	affects ipg_clk and hclk inputs of esdehc2	esdhc2_clk_enable
3	esdhc2_perclk	affects ipg_clk_perclk input of esdhc2	esdhc2_serial_clk_enable
4	esdhc3_ipg_hclk	affects ipg_clk and hclk inputs of esdehc3	esdhc3_clk_enable
5	esdhc3_perclk	affects ipg_clk_perclk input of esdhc3	esdhc3_serial_clk_enable
6	esdhc4_ipg_hclk	affects ipg_clk and hclk inputs of esdehc4	esdhc4_clk_enable
7	esdhc4_perclk	affects ipg_clk_perclk input of esdhc4	esdhc4_serial_clk_enable
8	ssi1_ipg	affects ipg_clk input of ssi1	ssi1_clk_enable
9	ssi1_ssi_clk	affects ccm_ssi_clk input of ssi1	ssi1_serial_clk_enable
10	ssi2_ipg	affects ipg_clk input of ssi2	ssi2_clk_enable
11	ssi2_ssi_clk	affects ccm_ssi_clk input of ssi2	ssi2_serial_clk_enable
12	ssi3_ipg	affects ipg_clk input of ssi3	ssi3_clk_enable
13	ssi3_ssi_clk	affects ccm_ssi_clk input of ssi3	ssi3_serial_clk_enable
14	ssi_ext1		ssi_ext1_clk_enable
15	ssi_ext2		ssi_ext2_clk_enable

Table 7-37. CCGR4 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	pata		pata_clk_enable
1	sim ipg clock	affects ipg_clk input to sim	sim_clk_enable
2	sim serial clock	affects ipg_perclk input to sim	sim_serial_clk_enable
3	—	**Note:** These clocks are no longer supported. For better power saving, it is recommended to clear CCGR4[13:6] bits - along with configuration of CCDR[18] and CLPCR[23] bits. Please refer to description of these bits for the recommended values.	—
4	—		—
5	—		—
6	—		—
7	SAHARA		sahara_clk_enable
8	rtic clocks	it is not possible to close rtic clocks during run mode. writting ‘00’ to this cgr will not affect the clocks of rtic.	rtic_clk_enable
9	ecspi1_ipg	affects ipg_clk input of ecspi1	ecspi1_clk_enable
10	ecspi1_perclk	affects ipg_clk_per input of ecspi1	ecspi1_serial_clk_enable
11	ecspi2_ipg	affects ipg_clk input of ecspi2	ecspi2_clk_enable
12	ecspi2_perclk	affects ipg_clk_per input of ecspi2	ecspi2_serial_clk_enable
13	cspi_ipg	affects ipg_clk input of cspi	cspi_clk_enable
14	srtc clocks		srtc_clk_enable
15	sdma clocks	affects also ahbs2axi1	sdma_clk_enable

Table 7-38. CCGR5 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	spba clocks		spba_clk_enable
1	GPU clocks		gpu_clk_enable
2	garb clocks		garb_clk_enable
3	vpu clocks		vpu_clk_enable
4	vpu reference clock		vpu_serial_clk_enable
5	IPU clocks		ipu_clk_enable

MCG bit index	Controlled Clocks	Comments	CCM output affected
6	ipmux1, ipmux2	Bit 12 affects ipmux1 via spare_output_1, and Bit 13 affects ipmux2 via spare_output_2. The default value of those bits is '11' - for the default case, CCM will generate '1' for both spare_output_1 and spare_output_2. If either one of those bits will be set to '0', then the corresponding output will be set to '0' when system enters WAIT mode. This will allow system to turn off the IPMUX clocks in WAIT mode and can be used in LP-APM to preserve the IPMUX power if no accesses are expected on the IP bus. In run mode the IPMUX clocks will not be closed and the settings of those bits will not affect RUN mode.	spare_output_1 and spare_output_2.
7	emi_fast	affects only fast clock of emi	emi_fast_clk_enable
8	emi_slow	affects only slow clock of emi	emi_slow_clk_enable
9	emi_int1	affects only int1 clock of emi	emi_int1_clk_enable
10	emi_enfc	affects only enfc clock of emi	emi_enfc_clk_enable
11	emi wrck clock	affects the wrck clock of emi	emi_clk_enable
12	gpc ipg clock		gpc_clk_enable
13	spdif0 clock		spdif0_clk_enable
14	spdif1 clock		spdif1_clk_enable
15	spdif ipg clock		spdif_clk_enable

Table 7-39. CCGR6 Register Mapping

MCG bit index	Controlled Clocks	Comments	CCM output affected
0	—	**Note:** These clocks are no longer supported. For better power saving, it is recommended to clear CCGR6[3:0] bits.	—
1	—		—
2	CSI mclk1	Clock root gated internally in CCM- no external clock enable output is generated.	
3	CSI mclk2	Clock root gated internally in CCM - no external clock enable output is generated.	
4	emi_garb	affects only garb clock to emi	emi_garb_clk_enable
5	IPU di0 clock		ipu_di0_clk_enable
6	IPU di1 clock		ipu_di1_clk_enable
7	gpu2d clock		gpu2d_clk_enable
8	Reserved		

MCG bit index	Controlled Clocks	Comments	CCM output affected
9	Reserved		
10	Reserved		
11	Reserved		
12	Reserved		
13	Reserved		
14	Reserved		
15	Reserved		

7.3.4.1 CCM Module Enable Override Register(CMEOR)

Figure 7-35 represents the CCM Module Enable Override Register (CMEOR). The CMEOR register contains bits to override the clock enable signal from the module. This should be used in case that it is decided to bypass the clock enable signals from the modules. In this case the word bypass is defined as the clock remaining on, unless gated by the corresponding CCGR bit. This bit will be applicable only for module that their clock enable signal is used provides its field descriptions.

73FD_4084 (CMEOR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1	1	1	1	1	1	1	1	mod_en_ov_emi_m7	mod_en_ov_emi_m6	mod_en_ov_emi_m5	mod_en_ov_emi_m4	mod_en_ov_emi_m3	mod_en_ov_emi_m2	mod_en_ov_emi_m1	mod_en_ov_emi_m0
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	mod_en_ov_emi_int1	mod_en_ov_emi_slow	mod_en_ov_emi_fast	mod_en_ov_emi_garb	1	mod_en_ov_gpu2d	mod_en_ov_vpu	mod_en_ov_dap	mod_en_ov_gpu	mod_en_ov_epit	mod_en_ov_gpt	mod_en_ov_esdhc	mod_en_ov_iim	mod_en_ov_owire	1	mod_en_ov_sahara
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 7-35. CCM Module Enable Override Register (CMEOR)

Table 7-40. CMEOR Field Descriptions

Field	Description
31–27	Reserved
26	reserved **Note:** Functionality is no longer supported. Bit shouldn't be altered and must remain in its default value.

Table 7-40. CMEOR Field Descriptions (continued)

Field	Description
25	reserved **Note:** Functionality is no longer supported. Bit shouldn't be altered and must remain in its default value.
24	reserved **Note:** Functionality is no longer supported. Bit shouldn't be altered and must remain in its default value.
23 mod_en_ov_emi_m7	overrides clock enable signal from emi_m7 - clock will not be gated based on emi's signal 'm7_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
22 mod_en_ov_emi_m6	overrides clock enable signal from emi_m6 - clock will not be gated based on emi's signal 'm6_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
21 mod_en_ov_emi_m5	overrides clock enable signal from emi_m5 - clock will not be gated based on emi's signal 'm5_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
20 mod_en_ov_emi_m4	overrides clock enable signal from emi_m4 - clock will not be gated based on emi's signal 'm4_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
19 mod_en_ov_emi_m3	overrides clock enable signal from emi_m3 - clock will not be gated based on emi's signal 'm3_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
18 mod_en_ov_emi_m2	overrides clock enable signal from emi_m2 - clock will not be gated based on emi's signal 'm2_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
17 mod_en_ov_emi_m1	overrides clock enable signal from emi_m1 - clock will not be gated based on emi's signal 'm1_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
16 mod_en_ov_emi_m0	overrides clock enable signal from emi_m0 - clock will not be gated based on emi's signal 'm0_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
15 mod_en_ov_emi_int1	overrides clock enable signal from emi_int1 - clock will not be gated based on emi's signal 'int1_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
14 mod_en_ov_emi_slow	overrides clock enable signal from emi_slow - clock will not be gated based on emi's signal 'slow_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal

Table 7-40. CMEOR Field Descriptions (continued)

Field	Description
13 mod_en_ov_emi_fast	overrides clock enable signal from emi_fast - clock will not be gated based on emi's signal 'fast_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
12 mod_en_ov_emi_garb	overrides clock enable signal from emi_garb - clock will not be gated based on emi's signal 'garb_clk_gate_en'. 0 dont override module enable signal 1 override module enable signal
11	Reserved
10 mod_en_ov_gpu2d	overrides clock enable signal from gpu2d - clock will not be gated based on gpu2d's signal 'gpu2d_busy'. 0 dont override module enable signal 1 override module enable signal
9 mod_en_ov_vpu	overrides clock enable signal from vpu- clock will not be gated based on vpu's signal 'vpu_idle'. 0 dont override module enable signal 1 override module enable signal
8 mod_en_ov_dap	overrides clock enable signal from dap- clock will not be gated based on dap's signal 'dap_dbgen'. 0 dont override module enable signal 1 override module enable signal
7 mod_en_ov_gpu	overrides clock enable signal from GPU. 0 dont override module enable signal 1 override module enable signal
6 mod_en_ov_epit	overrides clock enable signal from epit - clock will not be gated based on epit's signal 'ipg_enable_clk'. 0 dont override module enable signal 1 override module enable signal
5 mod_en_ov_gpt	overrides clock enable signal from gpt - clock will not be gated based on gpt's signal 'ipg_enable_clk'. 0 dont override module enable signal 1 override module enable signal
4 mod_en_ov_esdhc	overrides clock enable signal from esdhc - clock will not be gated based on esdhc's signals 'hclk_en', 'ipg_clk_en' and 'ipg_per_clk_en'. 0 dont override module enable signal 1 override module enable signal
3 mod_en_ov_iim	overrides clock enable signal from iim - clock will not be gated based on iim's signal 'iim_clk_en'. 0 dont override module enable signal 1 override module enable signal
2 mod_en_ov_owire	overrides clock enable signal from owire - clock will not be gated based on owire's signal 'owire_clk_en'. 0 dont override module enable signal 1 override module enable signal
1	Reserved
0 mod_en_ov_sahara	overrides clock enable signal from SAHARA. 0 dont override module enable signal 1 override module enable signal

7.4 Functional Description

This section describes clock generation.

7.4.1 External Low Frequency Clock—CKIL

The i.MX51 can use either a 32 kHz, 32.768 kHz, or a 38.4 kHz crystal as the external low frequency source. Throughout this chapter, the low frequency crystal is referred to as the 32 kHz crystal. CMOS input buffer with schmitt trigger feature is used as the receiver of 32 kHz clock. This clock source should be active all the time the i.MX51 is powered on.

The 32 kHz entering the CCM is referred as CKIL. It is synchronized to ipg_clk and supplied to modules that need it.

The 32 kHz clock also enters the FPM to produce the DPLL reference clock.

7.4.2 External High Frequency Clock CKIH and Internal Oscillator

An external crystal connected to the on-chip oscillator circuit is used to generate a reference for the three on-chip PLLs. The oscillator circuit frequency range is 22 to 27 MHz. For flexibility, inputs CKIH1 and CKIH2 may be used to supply special frequencies for the TV encoder, SPDIF, and others. as an alternate to the PLL-generated clocks. Because CKIH1 and CKIH2 pass through the CAMP (Clock Amplifier) modules, their frequency range is limited to the frequency range of the CAMP. See the Clock Amplifier Parameters section of the data sheet for specifications.

7.4.3 DPLLs (Digital Phase-Locked Loops) and Reference Clocks

There are three DPLLs in the i.MX51 project, as follows:

- PLL1 (typical functional frequency 800 MHz)
- PLL2 (typical functional frequency 665 MHz)
- PLL3 (typical functional frequency 216 MHz)

Each DPLL is controlled by a PLL-IP interface module, which can use the following options as DPLL reference clock:

- Output of OSC (typical functional frequency 24 MHz)
- Output of FPM (multiplied CKIL) (typical functional frequency 32.768 kHz)

7.4.4 PLL Clock Selector

A clock selector exists on each DPLL output of the 3 DPLLs. The clock selector is used to select between the gated clock output of the DPLL (dpgdck) and the divided by 2 gated clock output of the DPLL (dpgdck_2). Please refer to DPLL chapter for the description of the DPLL output and input pins.
The selection of dpgdck_2 is done by setting DPDCK02_EN bit (DPCTL[12]) in the corresponding DPLL-IP interface module.

When low range frequencies are required by the DPLL, such as 150Mhz, the divided output should be selected.

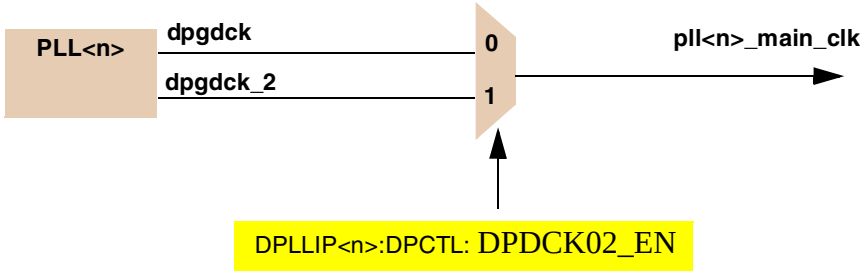


Figure 7-36. PLL clock selector

7.4.5 CLKSS Support

CLKSS input to the i.MX51 is internally connected to init_dp_ctl_dpllip[8] signal of each PLL-IP interface. This signal defines the initialization configuration of clock source to be used by PLL-IP interface:

- If CLKSS = 0, then OSC output is used as reference clock for all three DPLL_IPs.
- If CLKSS = 1, then FPM output is used as reference clock for all three DPLL_IPs.

During ignition process the value of clkss signal will be used in ccm to set either cosc_en bit or fpm_en bit, depending on which is the selected reference clock. The CAMP1_en and CAMP2_en will always be enabled when coming out of ignition process, no matter what was the value of clss. In run mode, software can control cosc_en, CAMP1_en, CAMP2_en and fpm_en by reprograming those bits.

7.4.6 CCM Internal Clock Generation

The clock generation is comprised of three submodules, as follows:

- CCM_CLK_SWITCHER
- CCM_CLK_IGNITION
- CCM_CLK_ROOT_GEN
- CCM_CLK_SWITCHER

CCM_CLK_SWITCHER sub module receives the pll output clocks and the pll bypass clocks. It generates the following three main switchable clocks to be delivered for CCM_CLK_ROOT_GEN:

- pll1_sw_clk (typical functional frequency 800Mhz - used to supply ARM platform)
- pll2_sw_clk (typical functional frequency 665Mhz - used to supply axi/ahb/ip buses clocks)
- pll3_sw_clk (typical functional frequency 216Mhz - used to supply serial clocks like usb, ssi, and so on.)

Figure 7-37 describes the generation of the three switchable clocks. It also includes the frequency switch control submodule that is responsible for the frequency change during DVFS scenario.

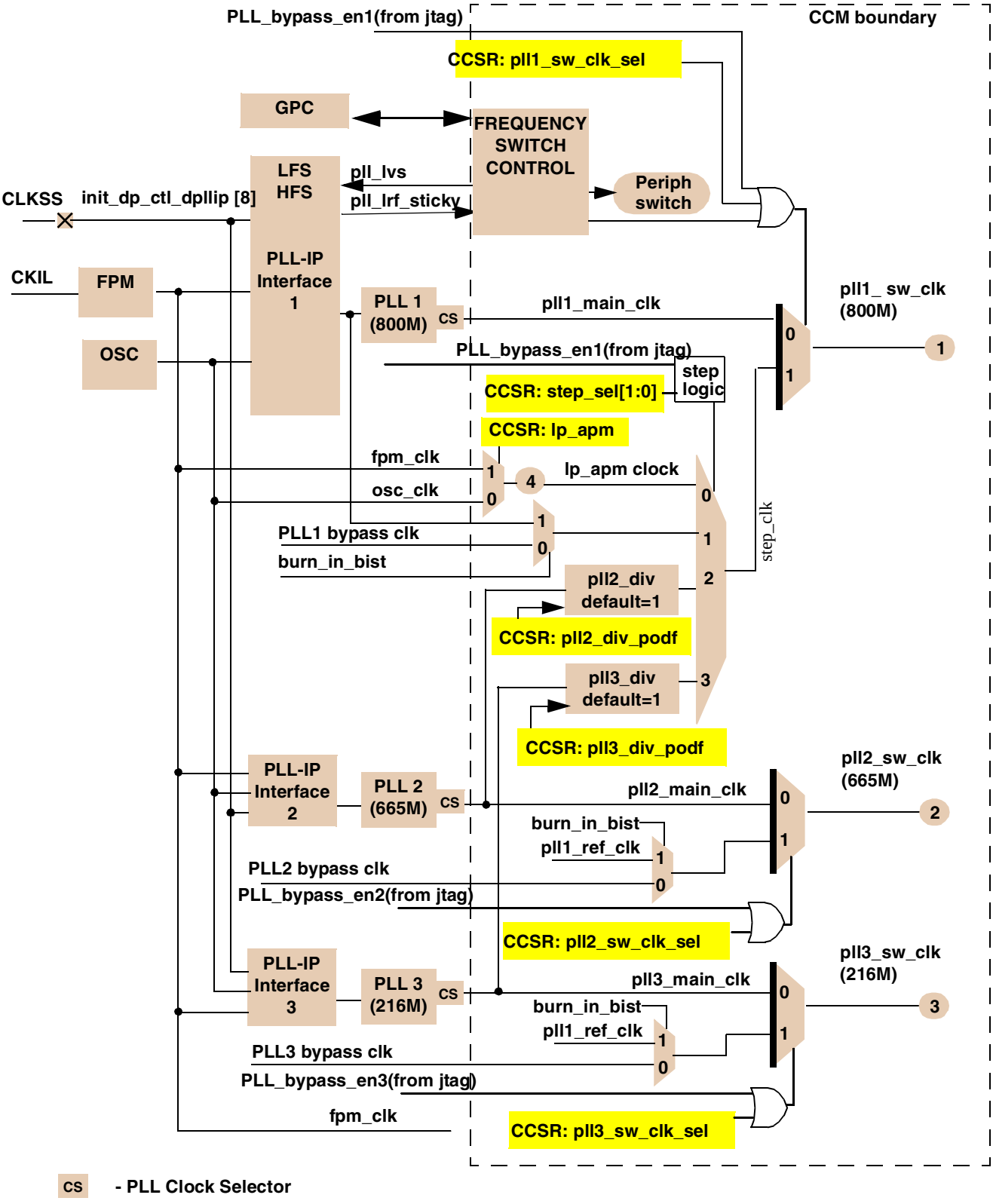


Figure 7-37. Switchable Clock Generation

7.4.6.1 PLL Bypass Procedure

CCM_CLK_SWITCHER sub module includes capability for each of the pll_main_clk's to be bypassed with an external bypass clk, and in this way supply to the three pll_sw_clk outputs a external bypass clock. In burn_in_bist mode the PLL bypass clocks for all three PLLs will be supplied from pll1_ref_clk (the output of DPLLIP1).

The decision to shift from pll_main_clk to pll_bypass_clk is done either by programable bits in CCSR or by three pll_bypass_en signals which are driven from jtag module. Since the switch between the clocks is done by a glitch less multiplexer, then both pll_main_clk and pll_bypass_clk should be running to complete the shift glitchlessly. In case the selection is done in reset, then it is not needed that both clocks will be available simultaneously.

NOTE

Since the bypass clock for pll1_sw_clk is generated from a multiplexer, the software needs to set the bypass clock via CCSR:step_sel[1:0] before shifting the glitchless mux via CCSR:pll1_sw_clk_sel. Similarly, when returning from pll_bypass, the system should first change the glitchless mux via CCSR:pll1_sw_clk_sel and only after that it is completed is change allowed to the CCSR:step_sel[1:0].

If the shift to pll_bypass for pll1_sw_clk is done via pll_bypass_en1 signal, then the pll_bypass_en1 signal should first set the step_select mux to prepare the bypass clock, and only after that to shift the glitch less mux from pll1_main_clk to the step_sel multiplexer output. This is taken care of since shift of step_sel mux is done faster than shift of the glitch less mux. When returning from pll_bypass to pll1_main_clk the step_sel mux control should be delayed until the glitch less mux has shifted back to the pll1_main_clk. This is taken care of in hardware by step_logic sub module.

7.4.6.2 Step Logic

The step logic allows the pll_bypass enable signal to bypass the step_sel bits and to set the 4x1 mux to the pll1_bypass_clk option. This sub module also delays the step_sel mux as described in PLL bypass procedure above.

7.4.6.3 DPLL-IP Reference Clock Connectivity

For all three DPLL-IPs, connections are as follows:

- The input clk2 is connected to on-chip oscillator clock output.
- The input clk3 is connected to FPM output (fpm_clkout).
- The input init_dp_ctl_dpllip [8] receives ccm_ipp_ind_clkss (this goes also to ccm).
- The input init_dp_ctl_dpllip [9] is connected to Vcc.

The above connections allows clkss to define in reset whether the FPM output or the on-chip oscillator output is chosen.

7.4.6.4 PLL Reference Clock Change

If software wants to change the PLL reference for a specific PLL, between FPM and on-chip oscillator outputs, or if software wants to stop a specific PLL, then software needs first to move all the clocks generated from this PLL to another PLL which is not changed. This should be done via the glitchless muxes for the clocks which cant be stopped (core and bus clocks). Then software should reprogram the respective dpll_ip: first configure DP_CONFIG[1], so that it will allow auto restart of the PLL on next update of the reference clock. Then software should change DP_CTL[8:9], to change the reference clock mux. In this case the respective PLL will restart and once its locked it will generate lrf_sticky signal to ccm. This signal can cause an interrupt from ccm (if configured in CIMR register). Based on this interrupt, software can be notified that the PLL is ready with the new reference clocks. Software should also take care of reprograming the PLL settings to match the new reference clock.At this point software can change the glitch less muxes to supply clock from the respective PLL that was changed.

7.4.6.5 CCM_CLK_IGNITION

The responsibility of the ccm_clk_ignition sub module is to manage the wake up after reset of the FPM, on-chip oscillator, CAMP1, CAMP2, dpll_ip and PLL modules. Its task starts with de-assertion of early_reset from the Reset controller. Its task finishes with assertion signal src_clock_ready that notifies reset controller that the root clocks are ready.

Figure 7-38 describes the connectivity of CCM clock ignition.

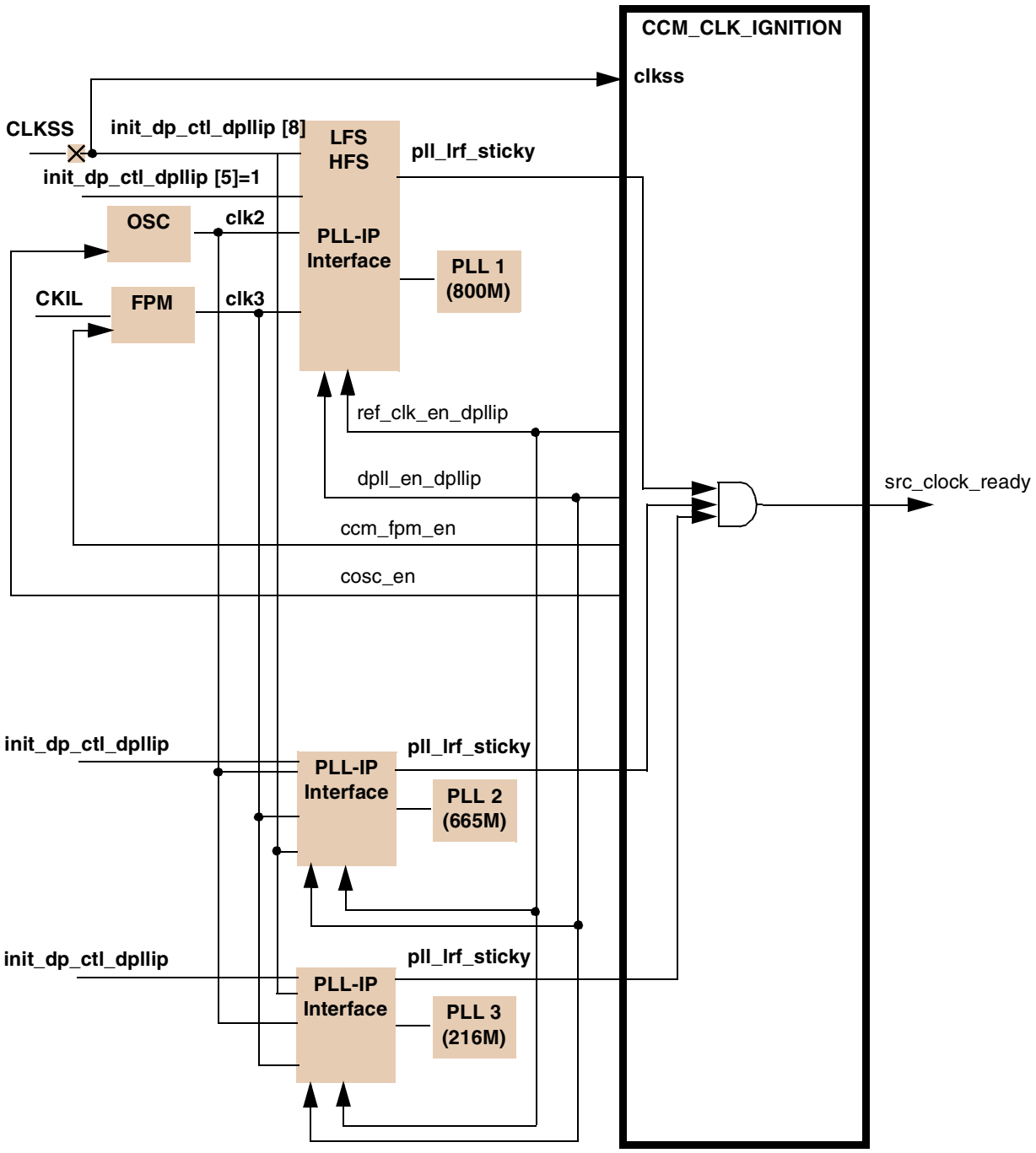


Figure 7-38. CCM Clock Ignition Connectivity

CLKSS is entering the three PLL_IP's to select upon reset which of the two clock sources to be used as reference clock for the PLL's. CLKSS is entering the init_dp_ctl_dpllip[8], while init_dp_ctl_dpllip[9] is constantly connected to '1'. This allows DPLL IP to choose between option 10 and 11 for dp_ctl[9,8]. It is not allowed to choose options 00 and 01 for dp_ctl[9,8], that is, it is not allowed to set dp_ctl[9] to '0'.

CLKSS is also entering the ccm_clk_ignition to notify which of the two sources (FPM or on-chip oscillator) is chosen as PLL source.

If FPM is chosen, fpm_en signal is asserted to start FPM. Next, ccm_clk_ignition should wait until FPM notifies by fpm_ready signal that its clock output is ready. In this stage CCM will assert the fpm_ready bit.

If the on-chip board is chosen, cosc_en signal will be asserted and cosc_pwrdown will remain deasserted to start the on-chip oscillator. Next ccm_clk_ignition will count the default number of CKIL defined in oscnt bits. Once the counter has elapsed, it is assumed that the clock output of on-chip oscillator is ready. In this stage CCM will assert the cosc_ready bit.

In both of the above cases, no matter what clkss chooses, the external oscillator and CAMPs will be enabled during ignition process: CAMP1_en=1, CAMP2_en=1 and ref_en_b=0 so that ckih_CAMP1 and ckih2_CAMP2 clocks will be started. Next, ccm_clk_ignition will count the default number of CKIL defined in oscnt bits. Once the counter has elapsed, it is assumed that the clock output of the CAMPs is ready. In this stage CCM will assert the CAMP1_ready and CAMP2_ready bit.

In case clkss defines on using the on-chip oscillator, then both oscnt counts for on-chip oscillator and external oscillator will be performed parallel, so that elapse of the count will notify that both oscillators are ready. This is done to save time during ignition process. If FPM is chosen, then both procedures of enabling FPM and counting oscnt value for the on-chip oscillator will start simultaneously.

Next, ccm_clk_ignition should assert ref_clk_en_dpllip so that reference clock to PLLs should be enabled.

Next, ccm_clk_ignition should assert dpll_en_dpllip so that dpllip_cpen will be generated by the DPLL-IP to the corresponding DPLL.

Note: For this, initial value of UPEN should be asserted 1, that is init_dp_ctl_dpllip[5] for all PLLs should be connected to 1.

Once the PLLs are locked, dpll_ip's will generate pll_lrf_sticky signal to ccm_clk_ignition.

Upon assertion of all three pll_lrf_sticky signals, ccm_clk_ignition will generate src_clock_ready signal for reset controller to indicate that PLL clocks are ready.

If pll_bypass_en is asserted, then src_clock_ready signal will be asserted as soon as exiting reset.

Upon arrival of src_clock_ready signal to reset controller, the reset sequence will continue. Please review SRC spec for details on this sequence.

7.4.6.6 Reset Values for DPLL-IP

Reset values that are hard coded to dpll-ip initialization values are as follows:

- PLL1—initial value = 262.144 MHz
- PLL2—initial value = 262.144 MHz
- PLL3—initial value = 229.376 MHz

These frequencies correspond to the reference clock source of CKIL (32 kHz), which generates a DPLL reference clock of 32.768 MHz (after multiplication of FPM by 1024).

These frequencies will be lower if the on-chip oscillator (24 MHz) is chosen as the source of DPLL. In that case the default frequencies generated are:

- PLL1—initial value = 208 MHz
- PLL2—initial value = 208 MHz
- PLL3—initial value = 182 MHz

7.4.6.7 CCM_CLK_ROOT_GEN

CCM_CLK_ROOT_GEN sub module generates the root clocks to be delivered to LPCG.

The following is a list of the root clocks generated by this module. The clocks are asynchronous unless otherwise stated.

- ARM_CLK_ROOT —generated from pll1_sw_clk.
- EMI_SLOW_CLK_ROOT
- eNFC_CLK_ROOT—generated by division of EMI_SLOW_CLK_ROOT and balanced with it.
- VPU_RCLK_ROOT
- AHB_CLK_ROOT
- IPG_CLK_ROOT—generated by division of AHB_CLK_ROOT and balanced with it.
- PERCLK_ROOT—this clock is synchronized and balanced to AHB_CLK_ROOT. For the synchronization process, perclk predivider and podf should generate clock 2.5 times lower than ahb clock. In peripheral DVFS scenario, when ahb is reduced, perclk should be configured to be lower 2.5 times the minimum value of ahb_clk (the DVFS value of ahb_clk).
- DDR_CLK_ROOT
- ARM_AXI_CLK_ROOT
- IPU_HSP_CLK_ROOT
- CKIL_SYNC_CLK_ROOT—CKIL clock synchronized to IPG_CLK_ROOT and balanced with it, when not in STOP mode. Synchronizer is bypassed, when in STOP mode.
- USBOH3_CLK_ROOT
- ESDHC1_MSPRO1_CLK_ROOT
- ESDHC2_CLK_ROOT
- ESDHC3_CLK_ROOT
- UART_CLK_ROOT
- SSI1_CLK_ROOT
- SSI2_CLK_ROOT
- SSI_EXT1_CLK—connected to external BGA contact
- SSI_EXT2_CLK—connected to external BGA contact
- USB_PHY_CLK_ROOT
- TVE_216_54_CLK_ROOT
- DI_CLK_ROOT
- SPDIF0_CLK_ROOT

- SPDIF1_CLK_ROOT
- CSPI_CLK_ROOT
- WRCK_CLK_ROOT
- LPSR_CLK_ROOT
- PGC_CLK_ROOT—generated as division of ipg_clk and balanced to it.

7.4.6.7.1 Special Considerations for Configuring PERCLK

In addition to ensuring that PERCLK remains at least 2.5 times slower than the AHB clock, certain steps need to be followed to ensure robust operation of PERCLK when reconfiguring the PERCLK clock source.

To properly configure the PERCLK clock source, the following steps are required:

1. In the CCGR registers, gate the clocks to all PERCLK-dependent modules (see Table 7-41 for a list of PERCLK-dependent modules).
2. Select the desired input clock for the PERCLK root clock (to be either source from the peripherals main source clock or the lp_apm clock source). Refer to the CMCBR register, perclk_lp_apm_sel bit.
3. Configure the perclk_pred1, perclk_pred2, and perclk_podf dividers to the desired setting. Refer to the CBCDR register for details.
4. In the CCGR registers, enable the desired clocks for the PERCLK-dependent module clocks.

Table 7-41 lists the PERCLK-dependent module clocks and the associated CCGR register.

NOTE

When configuring the PERCLK clock source, these clocks must be gated, however, other unused clock source may also be gated to minimize power consumption.

Table 7-41. PERCLK-dependent Module Clock Sources

PERCLK-dependent Module Clocks	Associated CCGR Register
uart1_perclk	CCGR1
uart2_perclk	
uart3_perclk	
i2c1 clocks	
i2c2 clocks	
epit1_highfreq	CCGR2
epit2_highfreq	
pwm1_highfreq	
pwm2_highfreq	
gpt_highfreq	
owire clocks	

Table 7-41. PERCLK-dependent Module Clock Sources (continued)

PERCLK-dependent Module Clocks	Associated CCGR Register
esdhc1_perclk	CCGR3
esdhc2_perclk	
esdhc3_perclk	
esdhc4_perclk	
sim serial clock	CCGR4
ecspi1_perclk	
ecspi2_perclk	

The following figures describe clock generation described earlier in this section. The frequencies shown in parentheses are the default typical frequencies. Glitchless muxes are indicated by muxes with thick line.

Figure 7-39 shows the ARMC_CLK_ROOT generation.

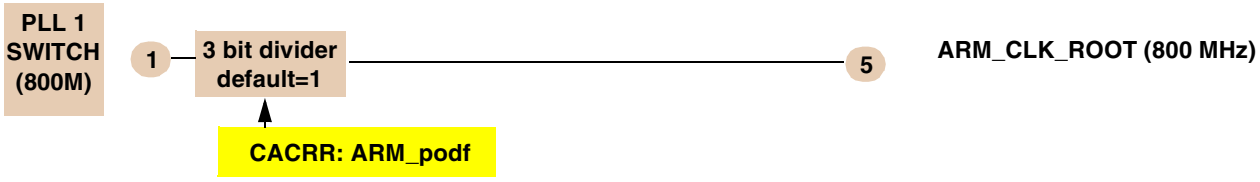


Figure 7-39. ARM_CLK_ROOT Generation

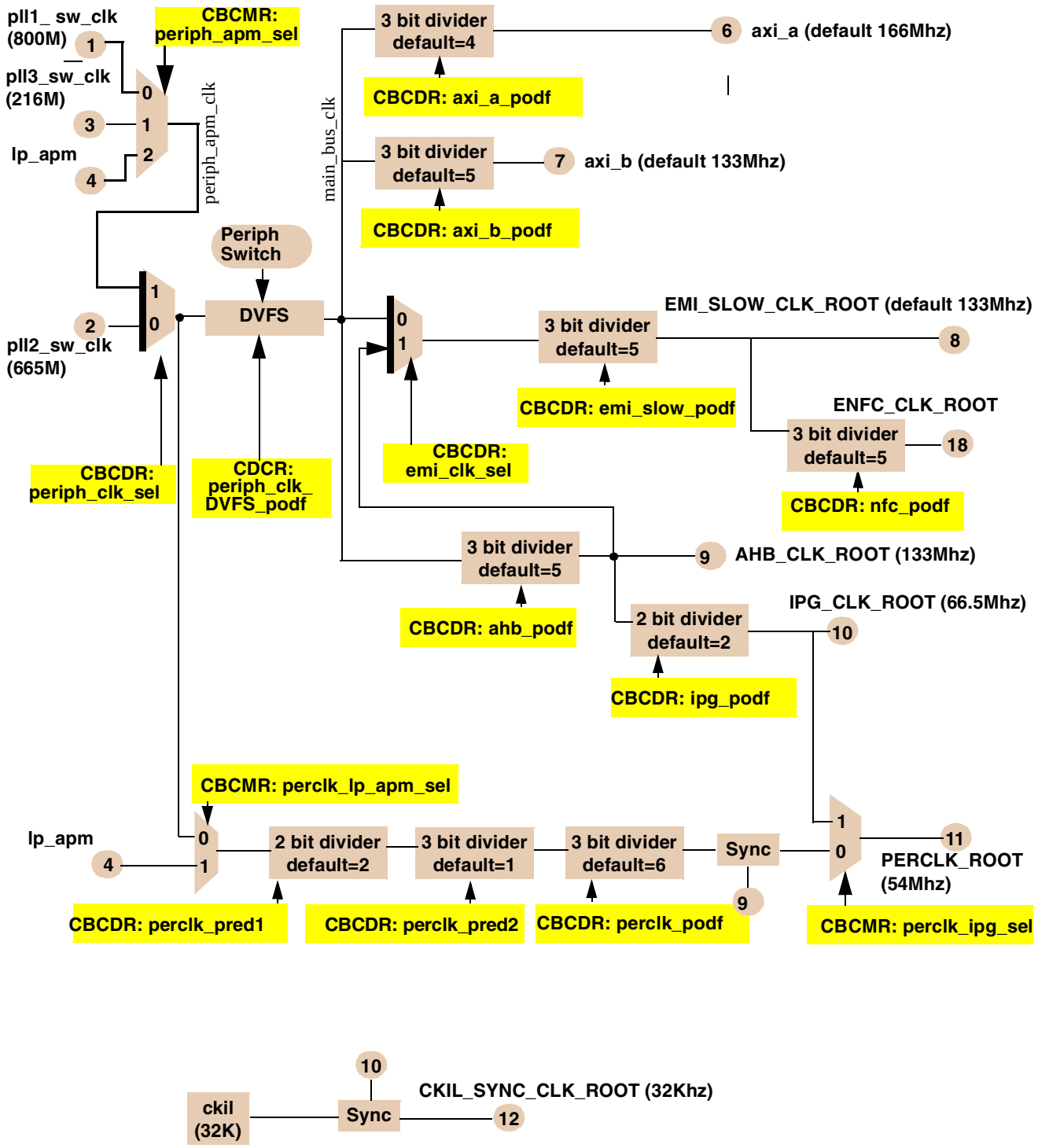


Figure 7-40. BUS clock generation

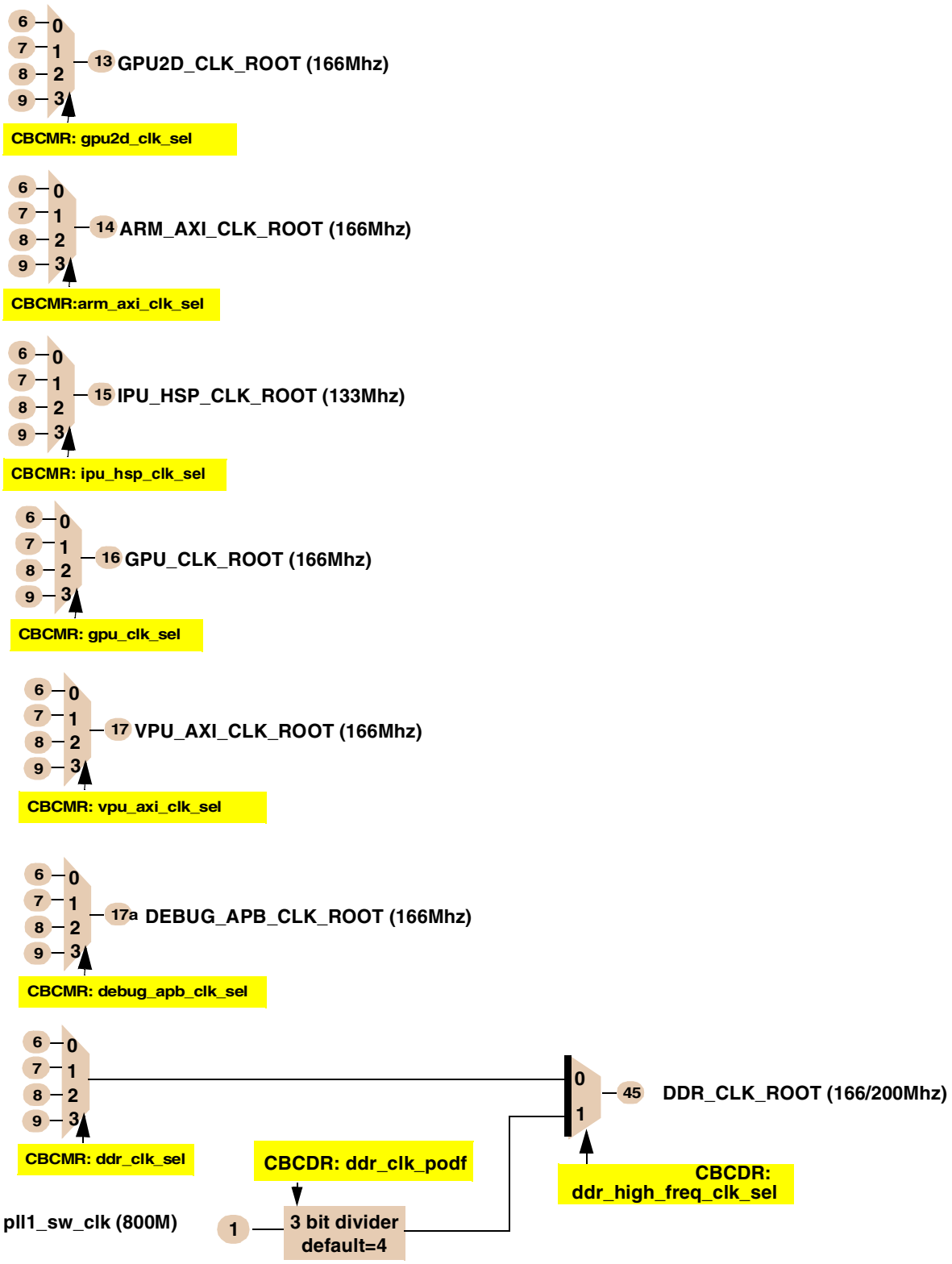


Figure 7-41. AXI Clocks Generation

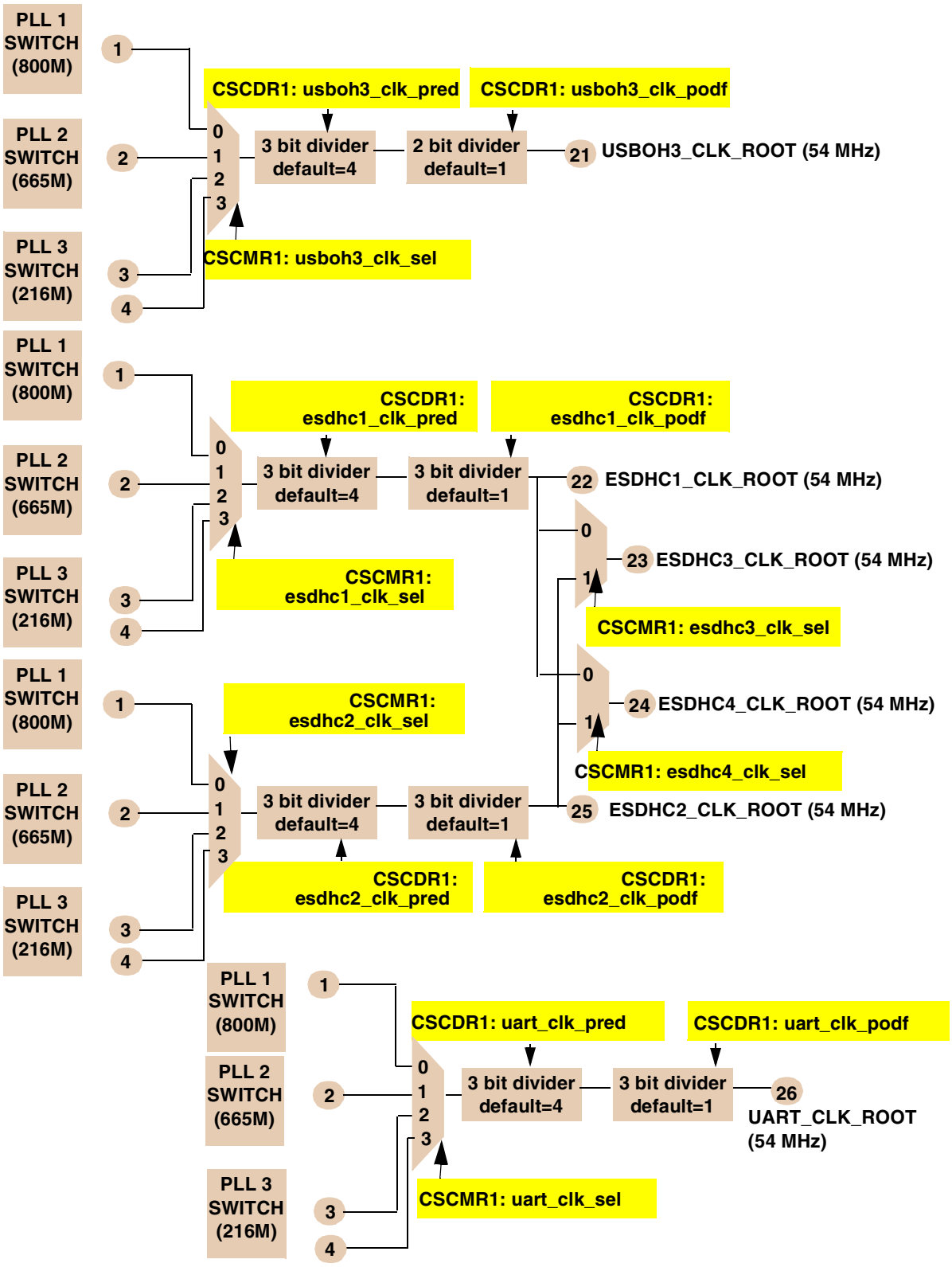


Figure 7-42. Serial Clock Generation (1 of 7)

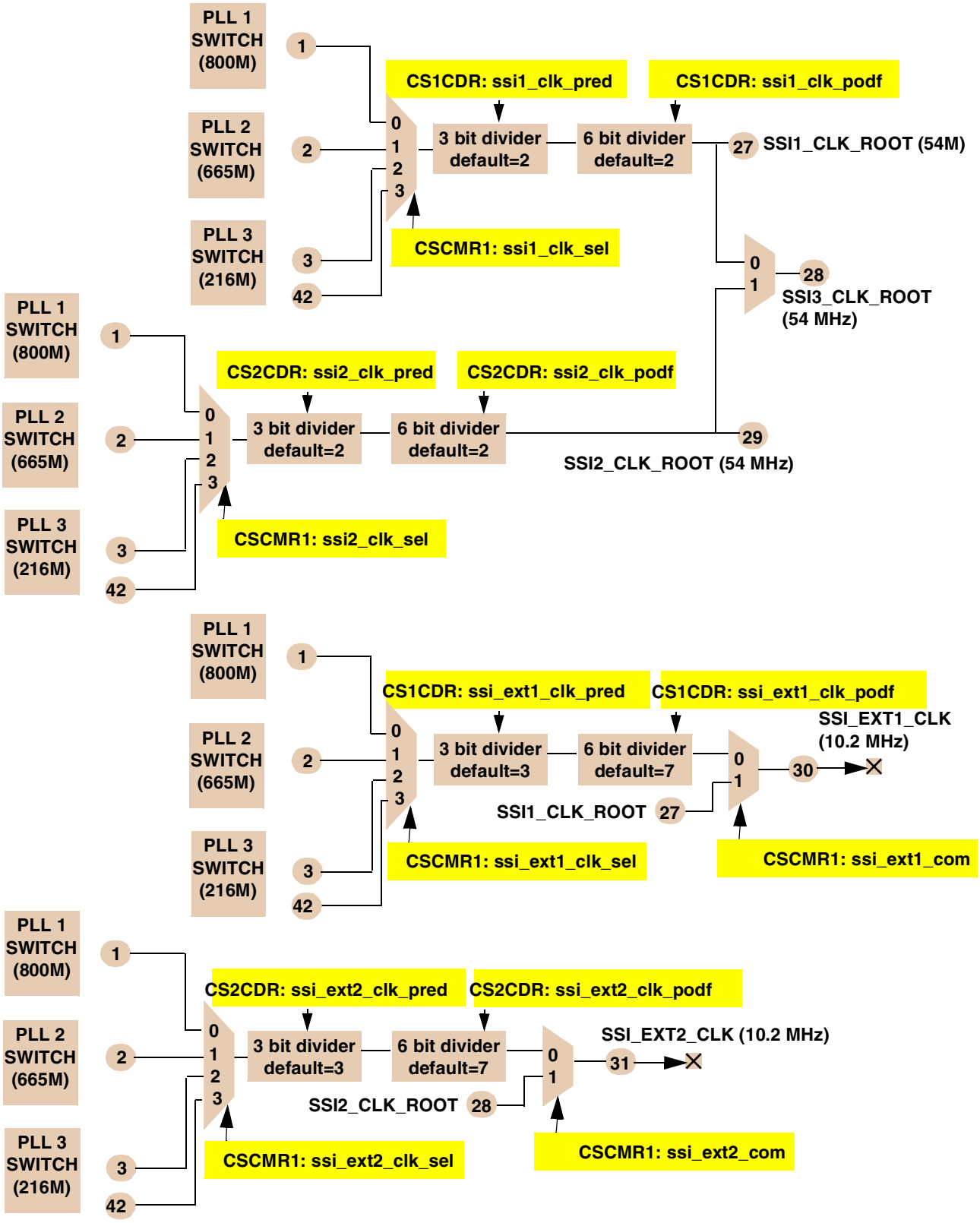


Figure 7-43. Serial Clock Generation (2 of 7)

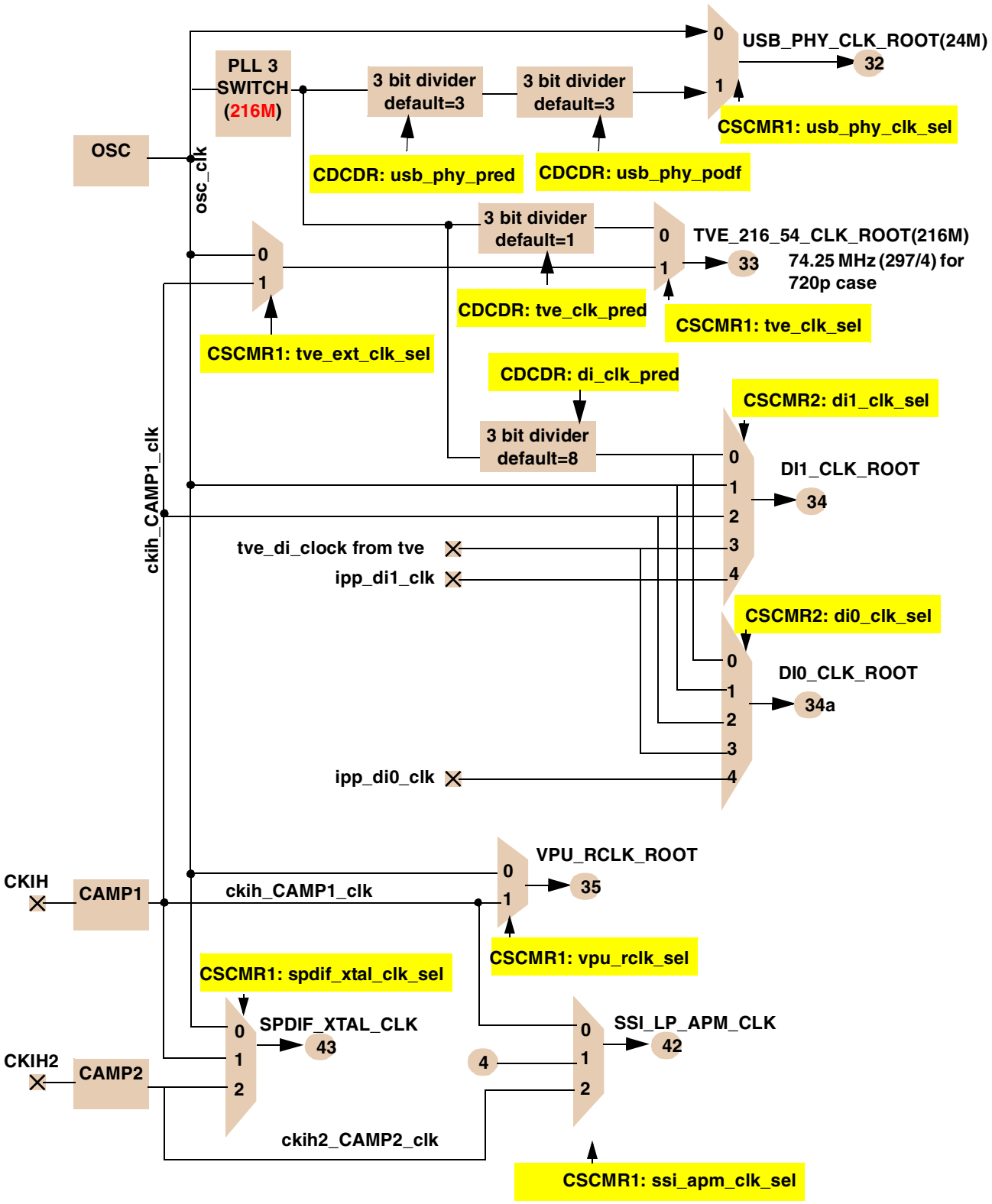


Figure 7-44. Serial Clock Generation (3 of 7)

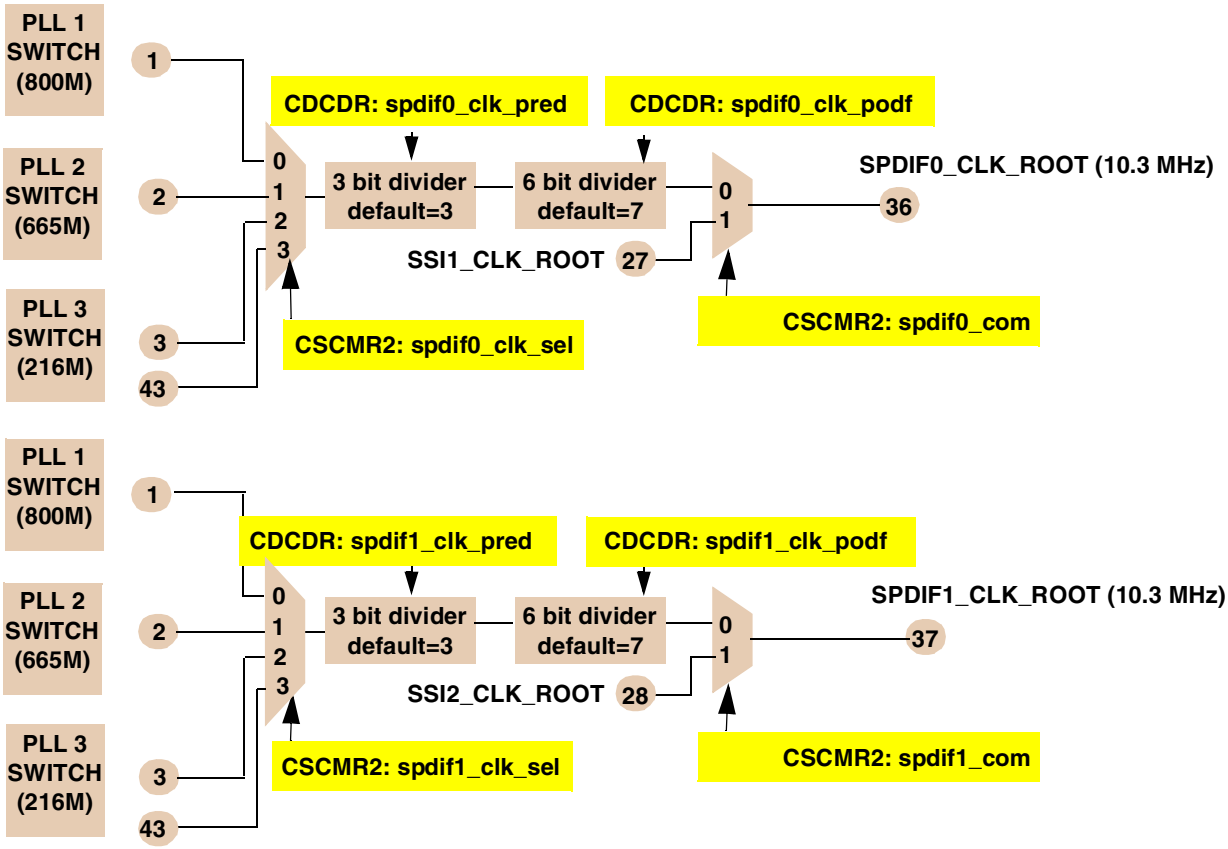


Figure 7-45. Serial Clock Generation (4 of 7)

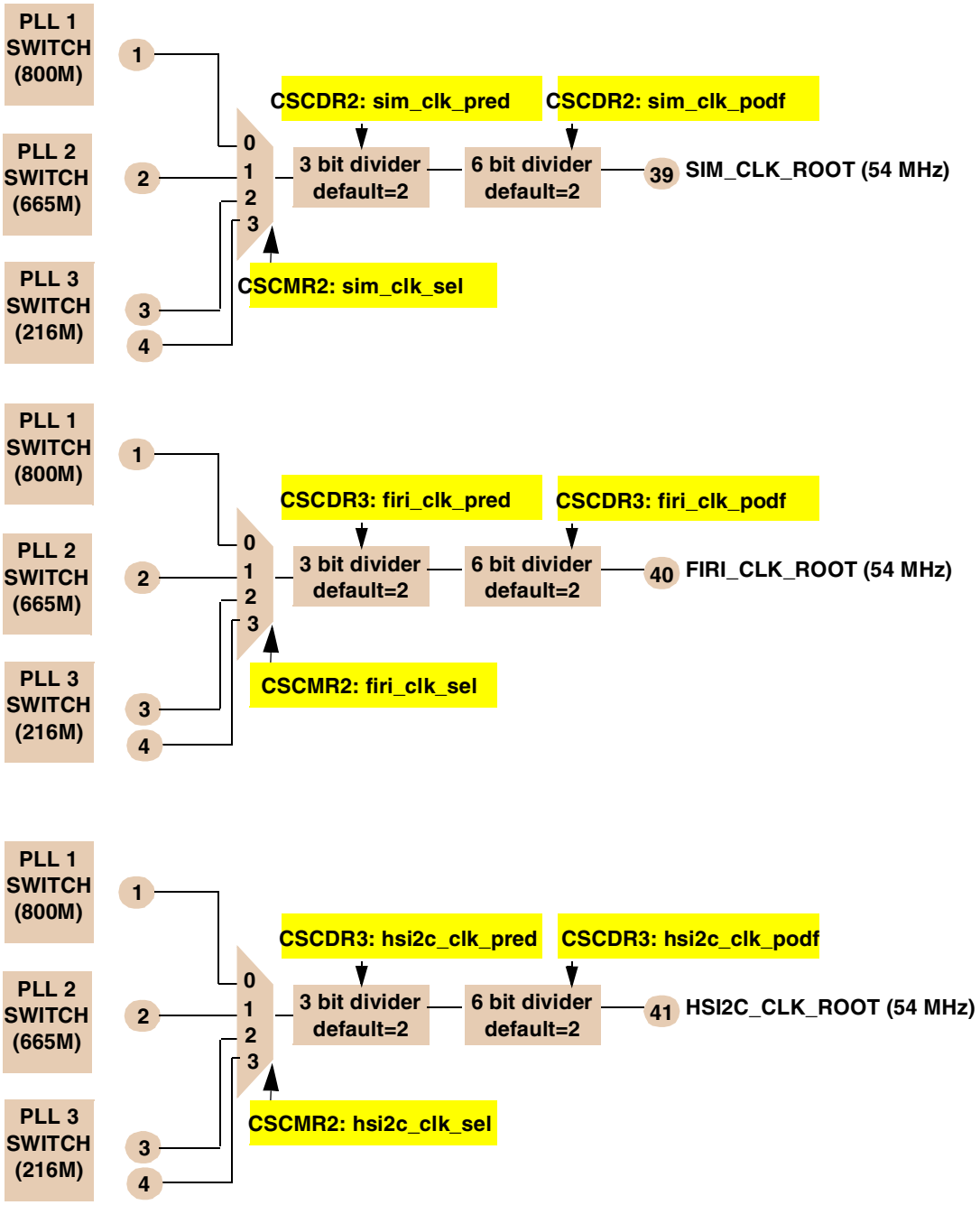


Figure 7-46. Serial Clock Generation (5 of 7)

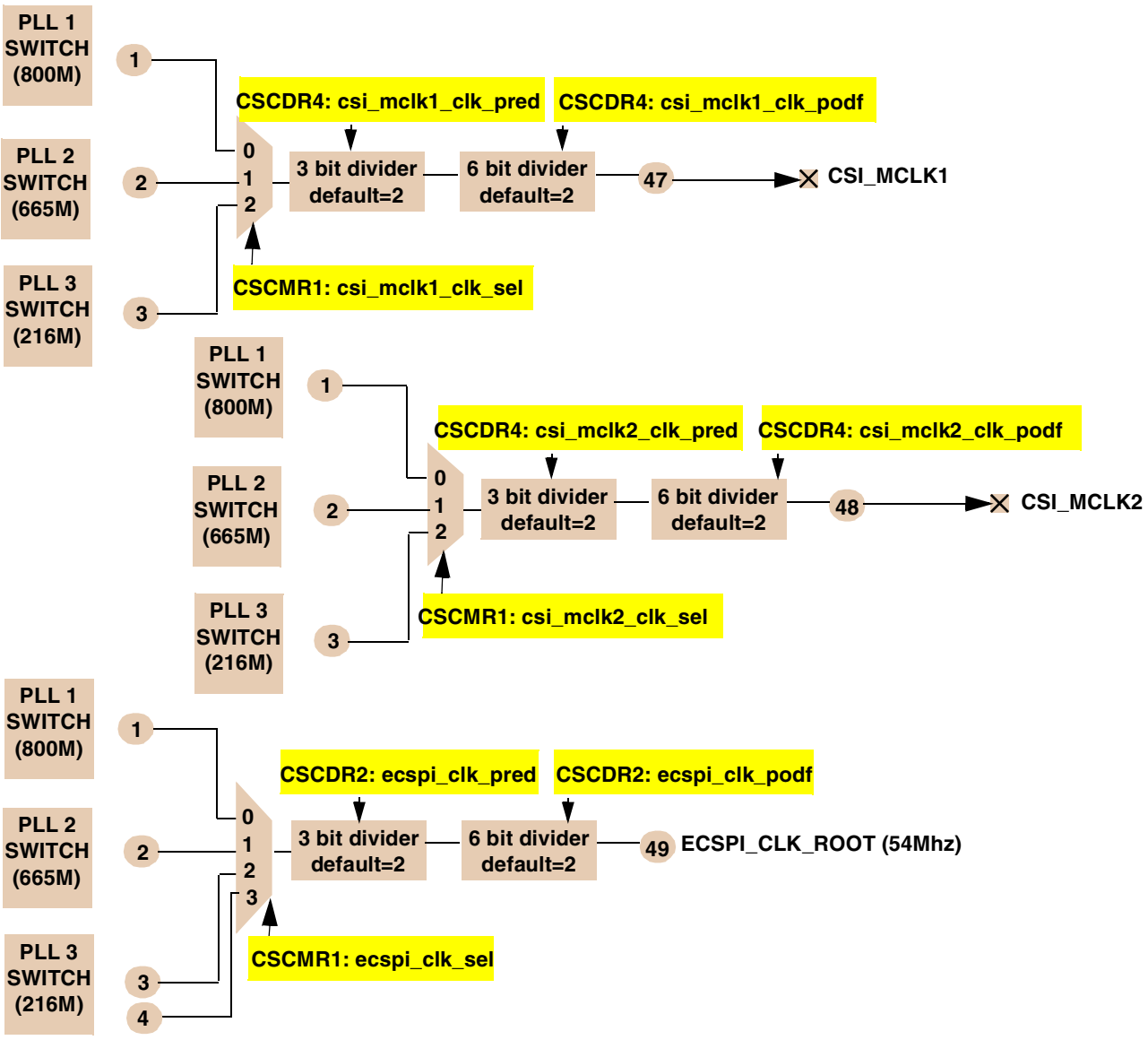


Figure 7-47. Serial Clock Generation (6 of 7)

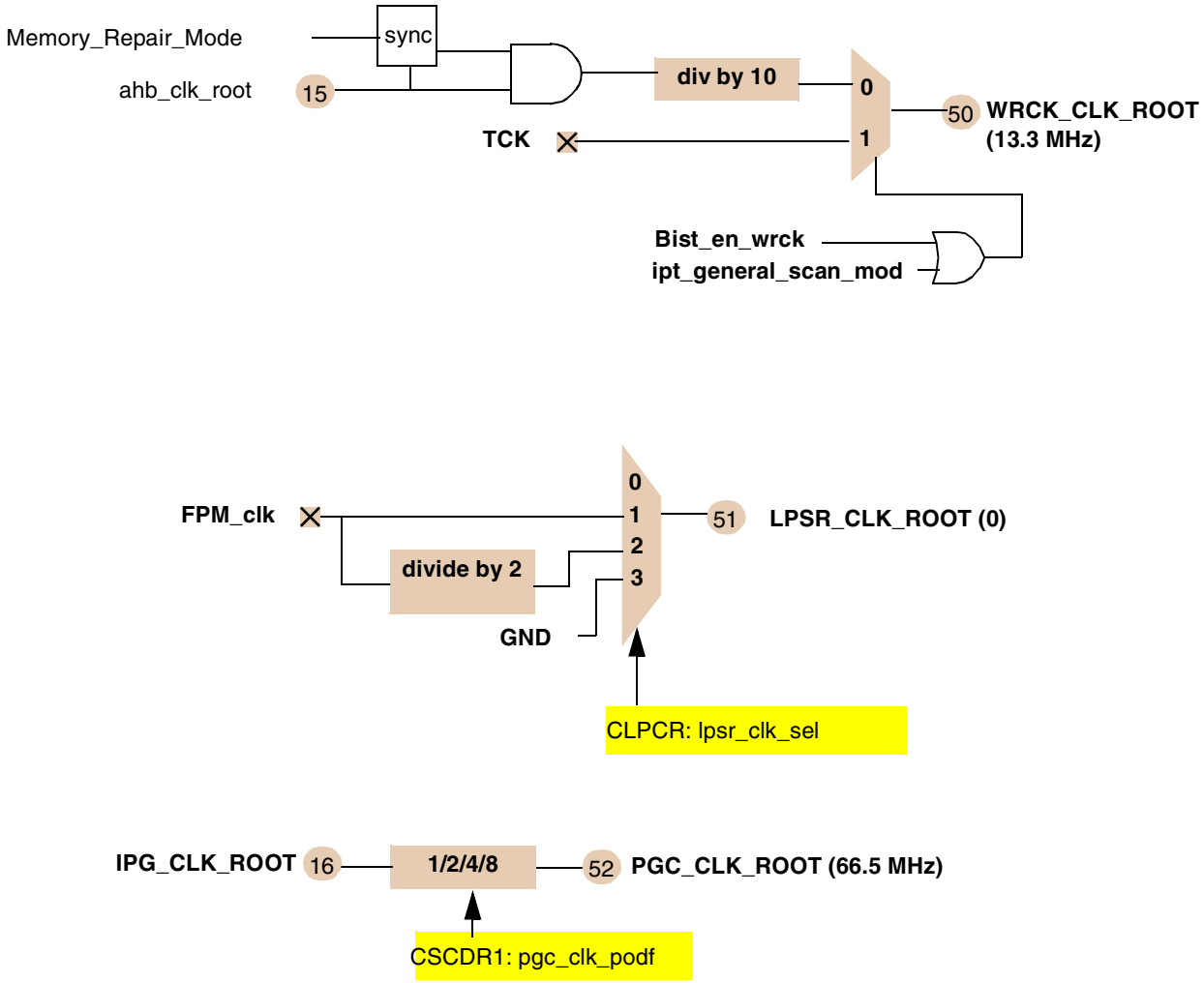


Figure 7-48. Serial Clock Generation (7 of 7)

NOTE

All 6-bit podf dividers in the previous figures are able to work on frequencies lower than the clock speeds defined in the i.MX51 datasheet. It is not allowed to insert the 6-bit divider a frequency higher than this frequency.

The following predividers should use divider values larger than ‘1’. The option ‘000’ is not allowed for them: usboh3_clk_pred, ssi1_clk_pred, ssi2_clk_pred, ssi_ext1_clk_pred, ssi_ext2_clk_pred, cspi_clk_pred.

7.4.6.7.2 Initial Values Controlled by SJC

The initial values of the following dividers and muxes can be controlled by SJC. In regular functional mode, the SJC will drive the reset values stated in the CCM register memory map. If SJC is progarmmed to change those values, then the reset value for those dividers/muxes will be taken from the SJC

programability. Software can update the changed reset value after reset sequence. The control signals and the dividers/muxes are listed below:

- init_perclk_pred1[1:0] to control reset value of perclk_pred1.
- init_perclk_pred2[2:0] to control reset value of perclk_pred2.
- init_perclk_podf[2:0] to control reset value of perclk_podf.
- init_ipg_podf[1:0] to control reset value of ipg_podf.
- init_ahb_podf[2:0] to control reset value of ahb_podf.
- init_axi_a_podf[2:0] to control reset value of axi_a_podf.
- init_axi_b_podf[2:0] to control reset value of axi_b_podf.
- init_emi_slow_podf[2:0] to control reset value of emi_slow_podf.
- init_nfc_podf[2:0] to control reset value of nfc_podf.
- init_periph_apm_sel[1:0] to control reset value of periph_apm_sel.
- init_periph_clk_sel to control reset value of periph_clk_sel.
- init_ssi_apm_clk_sel[1:0] to control reset value of ssi_apm_clk_sel.

7.4.6.7.3 Divider Change Handshake

Table 7-42 describes the dividers that may involve module handshake on every divider update. These handshakes should be bypassed if the respective module is disabled.

Table 7-42. i.MX51 Divider Handshake Summary

Divider	Handshake with module	Comment
emi_slow_podf	EMI IPU	Handshake with IPU will exist if emi_slow_podf is chosen as source to ipu_hsp_clk. Handshake with EMI will be through emi_DVFS_req_slow, emi_DVFS_req_int1,emi_DVFS_req_int2, emi_DVFS_ack_slow, emi_DVFS_ack_int1, emi_DVFS_ack_int2. If emi_slow_podf is chosen as source of gpu_clk, that is, the change of emi_slow_podf will affect gpu_clk then the handshake of emi_DVFS_req_garb, emi_DVFS_ack_garb should also commence. If this divider is chosen as source of ddr_clk,then the handshake of emi_DVFS_req_fast, emi_DVFS_ack_fast should also commence.
nfc_podf	EMI	Handshake with EMI will be through emi_DVFS_req_slow, emi_DVFS_ack_slow.
axi_a_podf	EMI IPU	Handshake with EMI will exist only if axi_a_podf is chosen as source to ddr_clk. In this case handshake with EMI will be through emi_DVFS_req_fast, emi_DVFS_ack_fast. If axi_a_podf is chosen as source of gpu_clk, that is, the change of axi_a_podf will affect gpu_clk then the handshake of emi_DVFS_req_garb, emi_DVFS_ack_garb should also commence.Handshake with IPU will exist only if axi_a_podf is chosen as source to ipu_hsp_clk.

Table 7-42. i.MX51 Divider Handshake Summary (continued)

Divider	Handshake with module	Comment
axi_b_podf	EMI IPU	Handshake with EMI will exist only if axi_b_podf is chosen as source to ddr_clk. In this case handshake with EMI will be through emi_DVFS_req_fast, emi_DVFS_ack_fast. If axi_b_podf is chosen as source of gpu_clk, that is, the change of axi_b_podf will affect gpu_clk then the handshake of emi_DVFS_req_garb, emi_DVFS_ack_garb should also commence. Handshake with IPU will exist if axi_b_podf is chosen as source to ipu_hsp_clk.
ahb_podf	EMI IPU	Handshake with EMI will exist only if ahb_podf is chosen as source to ddr_clk. In this case handshake with EMI will be through emi_DVFS_req_fast, emi_DVFS_ack_fast. If ahb_podf is chosen as source of emi_slow_clk generation, then similar handshake to the emi_slow_podf should take place. If ahb_podf is chosen as source of gpu_clk, that is, the change of ahb_podf will affect gpu_clk then the handshake of emi_DVFS_req_garb, emi_DVFS_ack_garb should also commence. Handshake with IPU will exist only if ahb_podf is chosen as source to ipu_hsp_clk.
ddr_clk_podf	EMI	Handshake is needed only if EMI's DDR (fast) clock is derived from this divider.
ddr_high_freq_clk_sel	EMI	Handshake of emi_DVFS_req_fast, emi_DVFS_ack_fast is needed.

Any update of CBCDR might involve handshake with modules based on the above table.

Once the CBCDR register is updated, CCM will check which of the above dividers was updated. For the updated dividers, CCM will check if a handshake with the respective modules is needed. If the handshake is needed, CCM will request acknowledge from the respective module for frequency change. Once the acknowledge arrives, CCM will perform the actual frequency change, and will notify the module that the frequency has changed by de asserting the request. IPU handshake also involves signal to notify it on the frequency change. This signal (periph_DVFS_sw_ack) is held for two clock cycles of ipg_clk on each change.

The handshake with the respective module will not be performed in case that the respective handshake is masked through CCDR[bits18–16]. In this case the write to the respective divider will commence immediately after the CBCDR register is updated.

Software has to make sure that the respective module is on and has the ability to acknowledge a request from the CCM. If the respective module is disabled or its clocks are gated off, then it will not have ability to support the handshake process and software should mask its handshake capability (through CCDR[bits18–16]).

Interrupts can be generated for each of the above dividers change. Please refer to CISR register for details on those interrupts.

In addition, CCM has divider handshake in process register (CDHIPR). This register has status bits for each of the above dividers that are involved in the handshake process. When the respective bit is asserted, it means that the respective dividers is being updated, and software should not attempt to write to this

divider until the respective bit is deasserted. Any reads of the respective divider during the time that the CDHIPR respective bit is asserted, will read the next value to be loaded to the divider, and not the actual dividers value. To make sure that software reads the actual dividers value, it should wait until the CDHIPR bits are desasserted.

All selections of muxes that are not glitchless should be updated only when there are no modules using their clocks, or the modules which use those clocks are disabled and their clocks are gated off.

Changing the periph_clk_sel bits in CBCDR will involve handshake with EMI and IPU - a similar handshake to the one that is performed in load dividers for EMI and IPU dividers. Those handshakes will be masked if configured to be masked in CCDR register".

Changing the ddr_high_freq_clk_sel bits in CBCDR will involve handshake with EMI - a similar handshake to the one that is performed in load dividers for EMI ddr. This handshake will be masked if configured to be masked in CCDR register.

Changing the emi_clk_sel bits in CBCDR will involve handshake with EMI and IPU - a similar handshake to the one that is performed in load dividers for EMI and IPU dividers. The IPU handshake will be performed only if IPU_HSP clock uses the emi_core_clk_root.

NOTE

In case DVFS is enabled (through GPC), it is not allowed to do any frequency changes, that is, the frequency of the system should be set prior to DVFS enable, and once the DVFS scenario is enabled, no divider of the above handshake group dividers should be changed "manually" by software since this might corrupt the handshake process.

7.4.6.7.4 CKIL Synchronizing to ipg_clk

CKIL is synchronized to ipg_clk when system is in functional mode. When system is in STOP mode, that is, when there is no ipg_clk, the CKIL synchronizer will be bypassed, and raw CKIL will be supplied to the system.

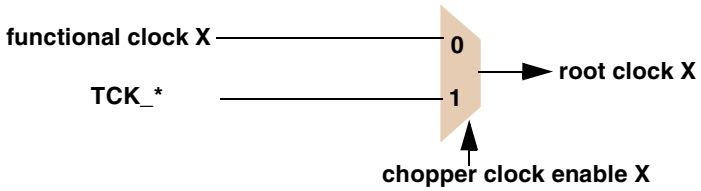


Figure 7-49. Chopper Control on Root Clocks

7.4.6.8 PLL Disabling/Enabling

PLL disabling and enabling is done via DPLL-IP module. Software should first move all the clocks generated from a specific PLL to another PLL, before disabling the PLL through DPLL-IP. This move of clocks can be done via glitch less mux for clocks which are critical to the system, that is, bus clocks. For

serial clocks, software should first disable the module and the clock generated to it, then move the mux controlling the source of the clocks to another PLL, and start back the module and its clocks. Only then it is safe to disable the PLL. The mux for the serial clocks is not glitch less hence the above procedure should be followed.

7.4.6.9 Observability Output signals

CCM has three muxes that generate critical signals to the IO PADS for observability. The three MUX outputs are connected to the three CCM outputs named obs_output_0, obs_output_1, obs_output_2. The three muxes are controlled via CTOR register of CCM.

It is possible to generate also the obs_input_0, obs_input_1, obs_input_2, obs_input_3, obs_input_4 and obs_input_5 to the external pads through the muxes controlled by CTOR. This allows to observe the signals connected to those 6 CCM input pins. Below list describes the signals connected for observability to those input pins:

- obs_input_0 connect to DPLLIP1 : dpllip_cpen
- obs_input_1 connect to DPLLIP1 : dpllip_cpres
- obs_input_2 connect to DPLLIP1 : dpllip_crstrt
- obs_input_3 connect to DPLLIP1 : dpllip_load_req
- obs_input_4 connect to DPLLIP2 : dpllip_cpen
- obs_input_5 connect to DPLLIP3 : dpllip_cpen

7.4.6.10 Low-Power Clock Gating (LPCG) Module

The LPCG module receives the root clocks and splits them to clock branches for each module. The clock branches are gated clocks. The enables for those gates can come from the following five sources:

- Clock enable signal from CCM—This signal is generated by configuration of the cgr bits in CCM. It is based on the low power mode.
- Clock enable signal from the module—This signal is generated by the module based on internal logic of the module. For clock enable signals from the module, that are used, CCM will generate override signal based on programable bit in CCM (CMEOR).
- Clock enable signal from Reset controller (SRC)—This signal will enable the clock during the reset procedure. Please refer to SRC chapter for details on the clock enable signal during reset procedure.
- Hard coded enable from fuse box.
- Enable or disable clock in BIST mode, based on bist_en signal. This will be applicable for clocks that are not functional and needed only on memory repair or bist sequence (like sms_clk’s and wrck).

The possible enable signals listed above are ANDed to generate the enable signal for the gating cell.

The enable signal for the gating cell is synchronized with the clock it needs to gate. This is done in order to prevent glitches on the gated clock.

Notifications are generated for the CCM, to indicate when to close and to open the clock roots. All notifications that correspond to the same clock root will be ORed to generate one notification signal to ccm for clock root gating.

Figure 7-50 describes the implementation for each gating cell:

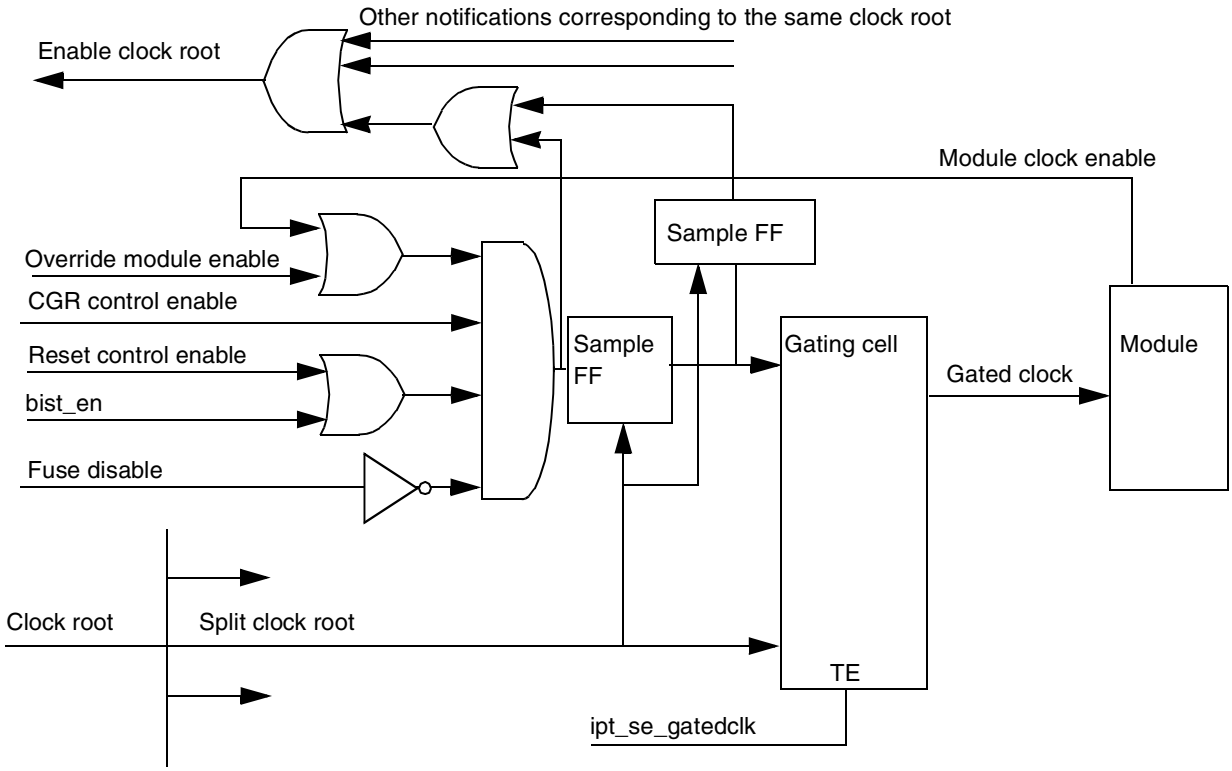


Figure 7-50. Gating Cell Implementation

Figure 7-51 describes the clock split inside LPCG module. It describes the case of two modules, one module without enable signal and one with enable signal. (SRC enable signals and sync FFs are omitted from this figure).

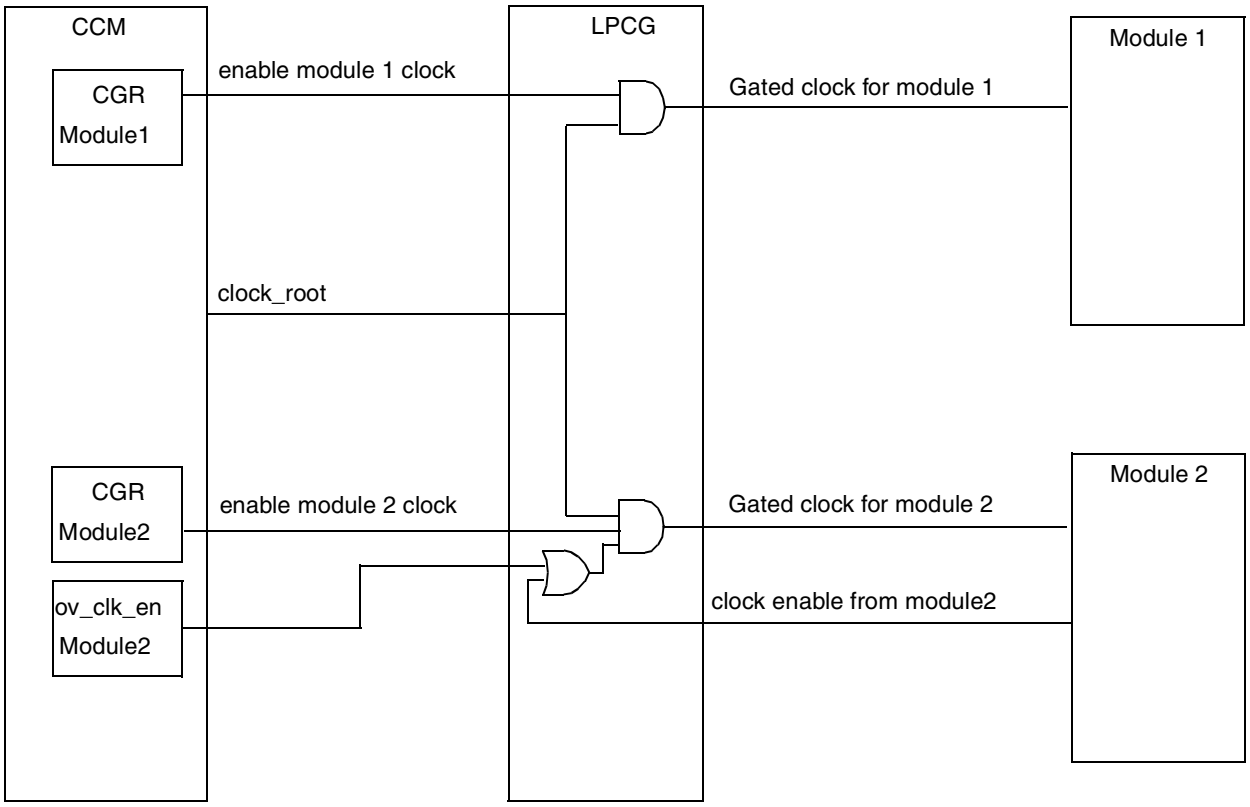


Figure 7-51. Clock Split in LPCG

7.4.7 DVFS Support

Frequency shift during DVFS procedure in the i.MX51 can be done on the following two domains:

- ARM clock domain frequency shift.
- Peripherals clock domain frequency shift (including buses).

The initiator of the frequency shift procedure is the GPC (Global Power Controller) module. It will initiate only one frequency shift at a time. There is no option to commence both of the above frequency shifts simultaneously. GPC will be aided by sdma and ccm to execute the frequency shift.

The next paragraphs explain the details of the above two frequency shift options.

7.4.7.1 ARM Clock Domain Frequency Shift

SDMA can request ARM frequency shift by using ARM clock divider or by relock of PLL1

The following steps will be configured by SDMA, (this can also be performed by ARM, but to be able to reduce load of ARM processing, it is preferable to perform it by SDMA):

1. SDMA can request ARM frequency shift by dividers, or by PLL1 relock. This request is written to CCM's arm_freq_shift_divider bit:
   - arm_freq_shift_divider = 0: PLL1 relock shift is requested.
   - arm_freq_shift_divider = 1: ARM_PODF divider shift is requested. In this case any new writes to ARM_PODF will be held until arm_clk_switch_req signal is asserted by GPC. This signal indicates that voltage is stable. The CCM holds a status bit to notify that the arm_podf divider is being in process of writing, and software should wait until the write finishes before writing any new value to arm_podf. Refer to CDHIPR register for details on the status bit. CCM can also generate interrupt once the actual arm_podf has been taken into effect in DVFS scenario. Please refer to CISR register for details.
2. Configure new ARM domain dividers (if arm_freq_shift_divider = 1: ARM_PODF divider shift is requested)

SDMA continues with the next steps only if (arm_freq_shift_divider = 0: PLL1 relock shift is requested).

3. Check which frequency mode is active in DPLLIP1, HFS or LFS: DPLLIP bit HFSM of DP_CTL register.
4. Configure the non active frequency mode (HFS or LFS) of DPLLIP1, with the new frequency needed for PLL1 to relock on: DPLLIP1 registers DP_OP, DP_MFD, DP_MFN, DP_HFS_OP, DP_HFS_MFD, DP_HFS_MFN.
5. Configure the step frequency divider: CCM CCSR register, bit pll2_div_podf or pll3_div_podf bits (depending on the decision in the step mux in step 6).
6. Configure the step frequency mux: CCM bit step_sel of CCM_CNT register.

Once those configurations are done, SDMA notifies GPC to continue with the frequency shift procedure.

The actual shift starts by assertion of arm_clk_switch_req signal from GPC to CCM.

CCM commences the following steps upon assertion of arm_clk_switch_req signal: These are hardware steps and do not require software involvement.

- If arm_freq_shift_divider = 1 dividers shift is requested):
  a) Load the ARM domain new dividers.
  b) After actual reloading of the dividers (all counters in 0), assert clk_switch_ack for 2 ipg cycles. This is done so GPC will have indication that frequency has been updated.
- If arm_freq_shift_divider = 0 (PLL1 relock shift is requested):
  a) Switch to the step frequency by a glitch less mux.
  b) After the glitchless mux has changed to the step frequency, invert pll_lvs signal to DPLLIP1 so that PLL1 will lock on the new frequency.
  c) On pll_lrf_sticky1 assertion move back to PLL1 that is locked on the new reloaded frequency by the glitch less mux.
  d) After actual shift of the mux assert clk_switch_ack for 2 ipg cycle. This is done so GPC will have indication that frequency has been updated.

Upon clk_switch_ack assertion, GPC deasserts arm_clk_switch_req signals.

At this point, ARM has changed frequency and system is ready for a new frequency switch procedure.

See Figure 7-52 for an illustration.

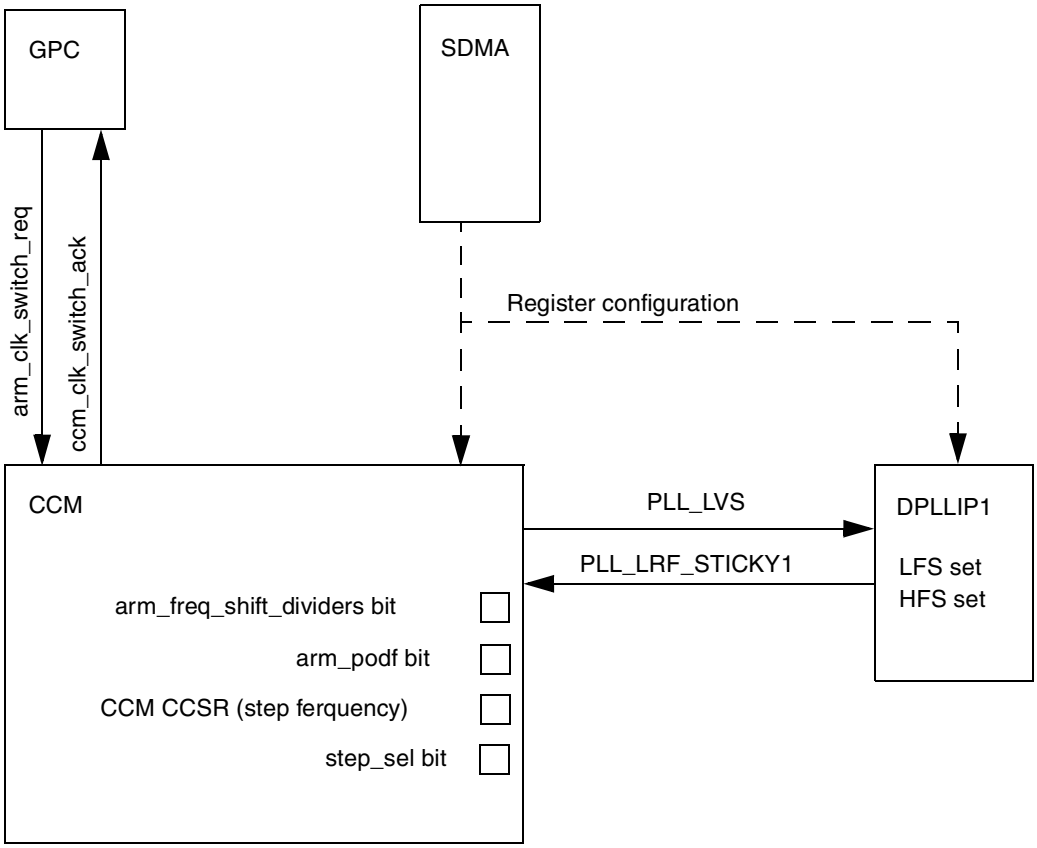


Figure 7-52. ARM DVFS Connectivity

7.4.7.2 Peripheral Clock Domain Frequency Shift

CCM includes capability to divide by 2/3/4 the root of bus clocks (ahb, axi, ipg, int_mem, ddr_clk, nfc_clk). The actual division factor is defined by bits in the ccm memory map, register CDCR [1–0]. The software is expected to set those bits prior to enabling DVFS operation. The value written to the DVFS register divider (CDCR[0–1]) will not be loaded immediately to the actual divider, but will wait until peripheral DVFS procedure will commence.

During DVFS scenario, the root clocks which are not affected by DVFS divider should be generated from a low frequency PLL. This frequency should not be higher than 240Mhz since the operating voltage is lowered in DVFS mode. Software can use the PLL3 with frequency 216Mhz for this operation. The setting for the serial clocks is done through CSCMR1 register.

The modules that their clock is about to change (IPU,EMI, and so on.) may need to be configured with values that determine the frequency of the new clock prior to the clock switch operation. Please refer to the respective module’s specification for clarification on the needed configuration.

Divide by 2/3/4 procedure (lowering frequency):

Upon assertion of periph_clk_div_req signal from GPC to CCM, the actual division procedure will take place. The periph_clk_div_req signal from GPC will remain asserted as long GPC requests a division of the peripheral clocks. This signal will be deasserted only when GPC requests to multiply the frequency and go back to the division by '1'.

Note: the DVFS operation can start also by software control using the sw_periph_clk_div_req. In that case the GPC control will be ignored and the ccm_clk_switch_ack will not be asserted.

There is no need to synchronize the three dividers change since they work on async clock domains.

Prior to the actual division there is need for acknowledge from peripherals that are sensitive to the frequency change. The modules sensitive to the frequency change are IPU and EMI. Those handshakes are described below:

- IPU handshake
  a) CCM asserts IPU_freq_change_req to request approval from IPU to commence the frequency shift.
  b) IPU asserts IPU_freq_change_ack to notify the time window that its ok from IPU point of view to change the frequency.

     Note that IPU does not support DVFS operation of division by 3.
- EMI handshake
  a) CCM asserts EMI_freq_change_req to request approval from EMI to commence the frequency shift.
  b) EMI asserts EMI_freq_change_ack to notify the time window that its ok from EMI point of view to change the frequency.

NOTE

Handshakes that are bypassed are not performed.

  c) CCM commences the actual division once all of the acknowledges (IPU_freq_change_ack, EMI_freq_change_ack) are asserted.
  d) After completing the division, CCM will notify the GPC by assertion of ccm_clk_switch_ack for 2 ipg cycles (rising on ipg_clk) and notify IPU by assertion of ipu_clk_changed for at least 2 hsp clock cycles (rising on the edge of the new hsp clock). CCM will also negate the IPU_freq_change_req and EMI_freq_change_req.

Multiply by 2/3/4 procedure (raising frequency): Upon negation of periph_clk_div_req signal from GPC to CCM, the actual multiplication procedure will take place by removing the division by 2/3/4 and returning to division by '1'.

Note: the DVFS operation can be stopped also by software control using the sw_periph_clk_div_req. In that case the GPC control will be ignored and the ccm_clk_switch_ack will not be asserted.

There is no need to synchronize the 3 dividers change since they work on async clock domains.

Prior to the actual frequency change there is need for acknowledge from peripherals that are sensitive to the frequency change. Those handshakes are described below:

- IPU Handshake
  a) CCM asserts IPU_freq_change_req.
  b) IPU asserts IPU_freq_change_ack to notify the time window if it is acceptable from the viewpoint of the IPU to change the frequency.
- EMI handshake
  a) CCM asserts EMI_freq_change_req.
  b) EMI asserts EMI_freq_change_ack to notify the time window that its ok from EMI point of view to change the frequency.

CCM performs the actual multiplication once both of the acknowledges are asserted.

After completing the frequency change, CCM notifies GPC by assertion of ccm_clk_switch_ack for 2 ipg cycles and notifies IPU by assertion of ipu_clk_changed for at least 2 hsp clock cycles. CCM also negates the IPU_freq_change_req and EMI_freq_change_req.

The following figure describes the connectivity.

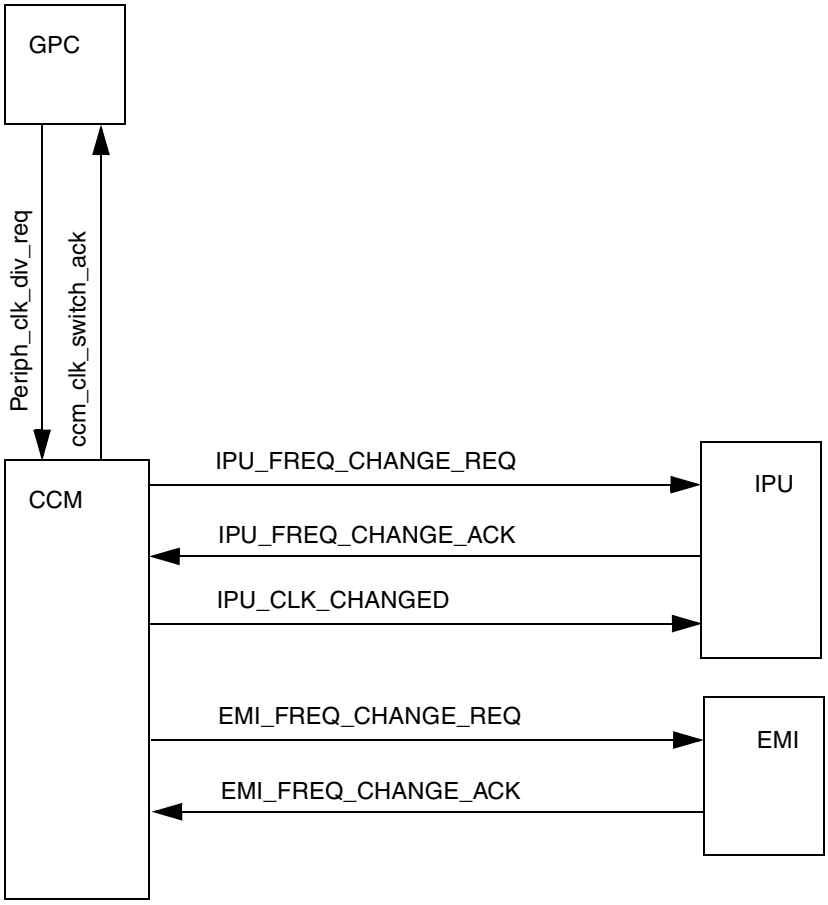


Figure 7-53. Peripheral DVFS Connectivity

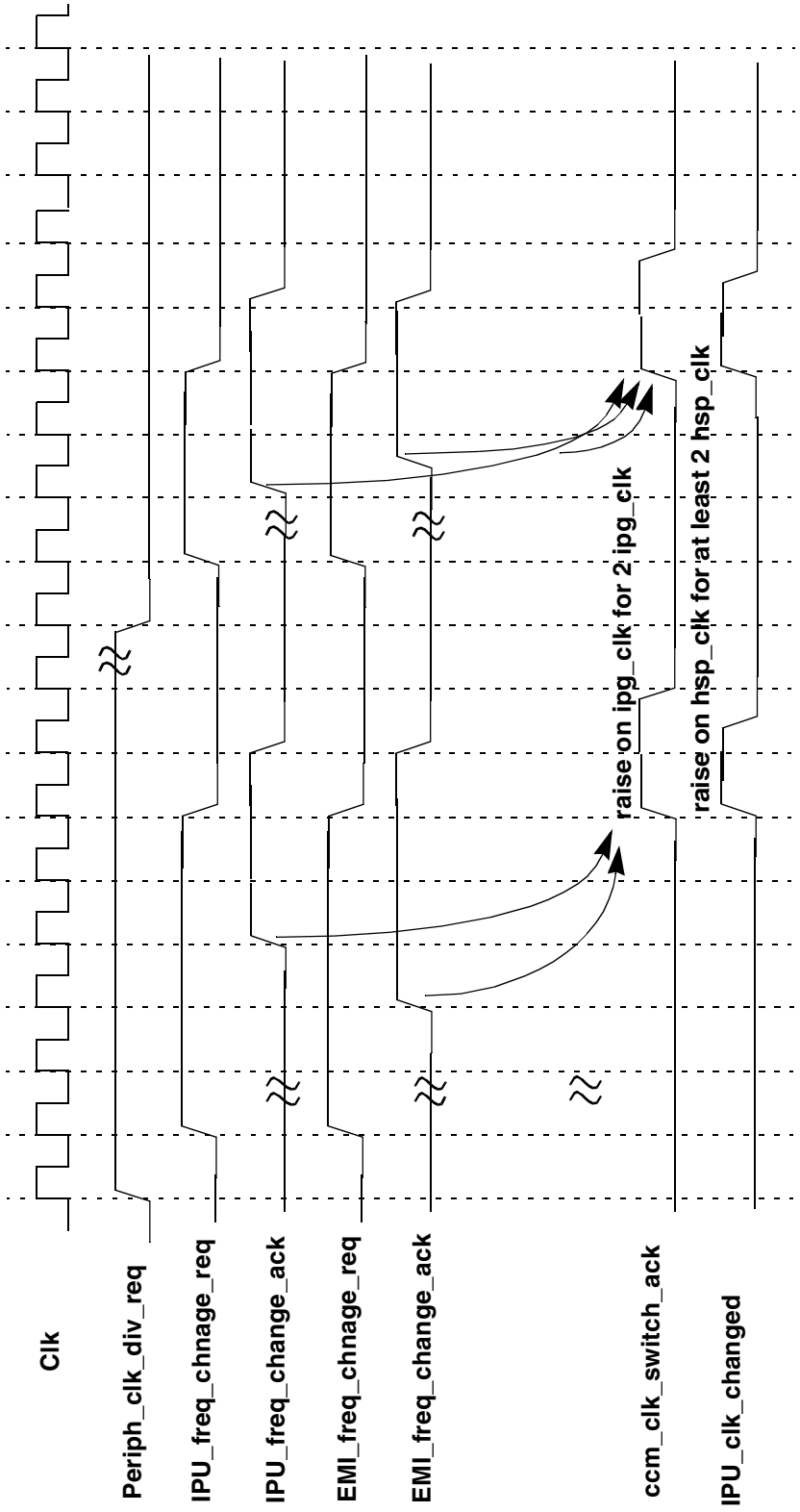


Figure 7-54. Peripheral DVFS Signals

7.4.7.3 Peripherals Restrictions in DVFS Scenario

On DVFS scenario for peripherals, IPG and AHB clocks are divided from their nominal frequency. The serial dividers in the low voltage case, should use a low frequency PLL input (below 300 Mhz) to work correctly.

Table 7-43 describes restrictions that need to be followed to support the DVFS frequency shift:

Table 7-43. DVFS Restrictions

Module	Support	Restriction
UART	R[1]	Need to configure uart rate to be lowest ipg clock frequency/16. Hence rates of 1.875M and 4M cannot be reached in DVFS scenario.
ESDHC	R	Need to configure card rate to be low_frequency_ahb_clk/2. Hence rate of 52 Mhz for SD card cannot be reached in the DVFS scenario.
MSHC	R	Need to configure card rate to be low_ipg_clk. Hence rated of 40M for card cannot be reach in DVFS scenario.
SSI	R	Master clock should be generated from ssi_ext1 and ssi_ext2 pads.
CSPI	R	Need to configure spi rate to be low ipg clk.
DDR memory	R	To be able to reach 200 Mhz on DDR, the clock can be generated from PLL1 that also supplies clock to ARM. In that case, PLL1 will have to be programmed to a multiplication of 200Mhz (200, 400, 600, 800) and the ARM DVFS should be enabled from divider change only and not from PLL change. If 166 Mhz is accepted for DDR memory, then the DDR clock should be generated from PLL2 (665 Mhz), and in that case ARM DVFS may be enabled.
USB, USB_PHY	N[2]	Needs ipg_clk>60 Mhz. Hence will not work under DVFS scenario
FEC	N	Needs ipg_clk>50 Mhz. Hence will not work under DVFS scenario
TVE	N	Is not synthesized for frequency operation
SPDIF	N	Is not synthesized for frequency operation y
P-ATA	N	Needs ipg_clk to be fixed. In case ARM core manages DVFS scenario, it can work only in times ipg_clk is fixed, that is, before and after DVFS scenario.

1 R= modules that will work with restriction

2 N = modules that will not work and need to be gated off if DVFS is used”

7.4.8 Power Modes

The i.MX51 supports four low-power modes: RUN mode, WAIT mode, STOP mode, LPSR (low power screen refresh) mode.

7.4.8.1 Run Mode

This is the normal/functional operating mode. In this mode ARM runs in its normal operational mode. The frequency and voltage can be changed upon DVFS scenario as described in Section 7.4.7, DVFS Support.” Clocks to the modules can be gated by configuring the corresponding cgr bits.

7.4.8.2 Wait Mode

In this mode the ARM clock is gated. All other clocks are functional and can be gated by programing their CGR bits. Power gating can be done on ARM platform.

7.4.8.2.1 Wait Mode Procedure

The wait mode procedure is as follows:

1. ARM writes to the LPM bit to set it for WAIT mode.
2. ARM writes to DSM_INT_HOLDOFF bit in TZIC.
3. DSM_INT_HOLDOFF assertion forces the interrupt controller to stop the synchronizer clock for incoming interrupts. Any interrupts that had previously been synchronized by the interrupt controller will remain asserted.
4. If a wakeup event occurred or an interrupt was serviced or is pending, the CPU may abort the DSM sequence by exiting the main shutdown code sequence and returning to the desired security mode.
5. If not, the software should execute the Wait For Interrupt (WFI) instruction.
6. Upon execution, the WFI instruction causes the ARM to drain its write buffers and enter a quiescent state. At this point the ARM asserts the STANDBYWFI signal.
7. The ARM platform then waits until the L2 cache controller has become inactive. Once the L2 is inactive and the CPU's STANDBYWFI output is asserted, the ARM platform asserts the ARM_DSM_REQUEST signal to the CCM.
8. The CCM continuously synchronizes both the ARM_DSM_REQUEST and TZIC_DSM_WAKEUP inputs into the clock domain used by its internal state machine.
9. If TZIC_DSM_WAKEUP is negated and ARM_DSM_REQUEST is asserted, the CCM begins the WAIT shutdown sequence:
10. CCM stops the ARM clock. (soc_mxclk input to ARM will be gated only if the respective CGR defines to gate it in WAIT mode).
11. Stop of ARM clock takes place only if CLPCR[5]=1 and debug_arm_clk_off_on_lpm=0 (connected to sjc register sjc_gpucr3_reg bit 15). If either CLPCR[5]=0 or debug_arm_clk_off_on_lpm=1 then ARM clocks will not be gated off and CCM will continue to next step.
12. CCM generates ccm_ipg_wait to indicate that it started WAIT entrance procedure.
13. CCM requests an acknowledge to close clocks of SAHARA,RTIC, IPU, SDMA, SCC, EMI and AHBMAX if their cgr bits indicate to close their clocks on WAIT mode and if their clocks was not already closed in run mode. The request is issued if the handshake is not bypassed by programing the CLPCR register bits 23–16. If the corresponding bits are set, the request signal is not issued to the corresponding module and CCM does not wait for its acknowledge in the process of entering low power mode. The requests commence via the following signals:
    - SAHARA: sahara_ipg_stop_req
    - RTIC: rtic_ipg_stop_req
    - IPU: ipu_stop_clk_at_wait_req
    - SDMA: sdma_ipg_stop_req

— SCC: scc_ipg_stop_req

— EMI: emi_lpmd, emi_lpmd_fast, emi_lpmd_int1, emi_lpmd_slow, emi_lpmd_garb - these requests will be asserted based on the definition of CCGRs. CCM will perform the appropriate EMI handshake based on the programing of the corresponding CCGR bits.

NOTE

Since EMI needs the aclk_fast (ddr clock) to be running for it to answer with emi_lpmd, then software should make sure to turn on the aclk_fast (ddr_clk) before entering any low power mode that requests the emi_lpmd.

— AHBMAX: ahbmax_halt_req

14. Once the above modules finished their operation and are ready to enter low power mode, they acknowledge CCM that its safe to turn of their clocks. The acknowledge signals are as follows:
    — SAHARA: sahara_ipg_stop_ack
    — RTIC: rtic_ipg_stop_ack
    — IPU: ipu_stop_clk_ack
    — SDMA: sdma_ipg_stop_ack
    — SCC: scc_ipg_stop_ack
    — EMI: emi_lpack, emi_lpack_fast, emi_lpack_int1, emi_lpack_slow, emi_lpack_garb
    — AHBMAX: ahbmax_halt_ack

NOTE

The SCC clocks are closed in WAIT mode only if SAHARA clocks are also closed, hence the SCC handshake is performed only if the SAHARA clocks are programmed to be closed as well.

15. The requests are generated by CCM in stages. CCM continues from stage to stage once all the acknowledgements of a present stages were received. The stages are the following:
    — Stage 1: SAHARA, RTIC, IPU, SDMA
    — Stage 2: SCC
    — Stage 3: AHBMAX
    — Stage 4: EMI
16. Once CCM receives all the acknowledge signals needed, and if at least 8 ipg_clk cycles have elapsed since the assertion of ccm_ipg_wait signal, it enters WAIT mode and does the following:.
    a) Closes the clocks to the modules which were defined to be shut at WAIT mode in the CCGR bits.
    b) Asserts system_in_wait_mode signal. This signal is connected to the IO pads for observability, to indicate of a WAIT mode.

Once CCM is in WAIT mode, it checks whether TZIC_DSM_WAKEUP signal has asserted or ARM_DSM_REQUEST has negated during the process of entering WAIT mode. If it has occurred, then CCM exits WAIT mode.

If TZIC_DSM_WAKEUP is not asserted and ARM_DSM_REQUEST is still asserted, the CCM moves to state SRPG_ARM and generates a request to GPC to power down the ARM platform, IPU, VPU,GPU, GPU2D and EMI by asserting signals ccm_pdn_4arm_req and ccm_pdn_4all_req. If in GPC those modules are defined to be powered down on WAIT mode, then GPC commences powering down for them. Please refer to CCM-GPC Connectivity diagram. The GPC sends a pdn_ack at the end of the power down sequence.

CCM's low power state machine remains in state "SRPG_ARM" until WAIT mode is exited.

Note that during WAIT mode, if SCC asserts either hclk_en signal or ipg_clk_en signal, the CCM asserts scc_clk_enable, thus enabling the SCC clocks. The WAIT mode is not exited due to this assertion.

7.4.8.2.2 Wait Mode is Exited by the Following Procedure

As soon as the synchronized TZIC_DSM_WAKEUP signal is seen as asserted or the ARM_DSM_REQUEST is negated, the CCM begins the process of exiting WAIT mode.

- Enable VPU, GPU2D and GPU clocks only.
  a) CCM requests that GPC restores power. CCM asserts ccm_pup_req to request that GPC powers up all the modules that were powered down if it was powered down on the entrance to WAIT mode.
  b) GPC notifies CCM by asserting signal PUP_ACK that power of ARM, VPU, IPU, and EMI is back on, and it's safe to exit from WAIT mode. Only then does CCM do the following:
     – Once assertion of notification from src that the resets for the power gated modules has been finished, (src_power_gating_reset_done is set) negate the low power request signals to all modules and enable all modules clocks including ARM clocks and return to run mode. (the clocks that their CCGR bits define not to be open in RUN mode will not be opened, and will continued to be gated in RUN mode).
     – Once system is in run mode, CCM will negate ccm_ipg_wait and system_in_wait_mode.
  c) Once the interrupt propagates through all the synchronization logic, the ARM CPU recognizes and services it. This forces the negation of the ARM_DSM_REQUEST output.

     Prior to this point, ARM_DSM_REQUEST and TZIC_DSM_WAKEUP were simultaneously asserted. Since TZIC_DSM_WAKEUP has top priority, the system is able to wake up.
  d) Before the ARM exits the ISR, it clears the interrupt source. This causes the TZIC_DSM_WAKEUP signal to be negated. At this point, the two low power control signals (TZIC_DSM_WAKEUP and ARM_DSM_REQUEST) are negated, and the WAIT sequence can be repeated at any time.

7.4.8.3 Stop Mode

In this mode all system clocks are stopped. PLLs are stopped. Power gating can be done on ARM platform, IPU, VPU and EMI. Synchronization of the CKIL clock is bypassed.

7.4.8.3.1 Entering Stop Mode

Stop Mode is entered by the following procedure:

1. ARM writes to LPM bit to set it for STOP mode.
2. ARM writes to DSM_INT_HOLDOFF bit in TZIC.
3. DSM_INT_HOLDOFF assertion forces the interrupt controller to stop the synchronizer clock for incoming interrupts. Any interrupts that had previously been synchronized by the interrupt controller will remain asserted.
4. If a wakeup event occurred or an interrupt was serviced or is pending, the CPU may abort the DSM sequence by exiting the main shutdown code sequence and returning to the desired security mode.

   If not the software, should execute the Wait For Interrupt (WFI) instruction.
5. Upon execution, the WFI instruction causes the ARM to drain its write buffers and enter a quiescent state. At this point the ARM asserts the STANDBYWFI signal.
6. The ARM platform waits until the L2 cache controller has become inactive. Once the L2 is inactive and the CPU's STANDBYWFI output is asserted, the ARM platform asserts the ARM_DSM_REQUEST signal to the CCM.
7. The CCM continuously synchronizes both the ARM_DSM_REQUEST and TZIC_DSM_WAKEUP inputs into the clock domain used by its internal state machine.

   The only time when those signals are not synchronized is when PLLs are closed. In this case exiting from STOP mode is done asynchronously (please refer to Section 7.4.8.3.2, Exiting Stop Mode").

If TZIC_DSM_WAKEUP is negated and ARM_DSM_REQUEST is asserted, the CCM begins the following STOP shutdown sequence:

1. CCM stops the ARM clock.
2. CCM generates ccm_ipg_stop and ccm_int_mem_ipg_stop to indicate that it started stop entrance procedure.
3. CCM requests an acknowledge to close clocks of SAHARA, RTIC, IPU, SDMA, SCC, EMI and AHBMAX if their clocks was not already closed in run mode. The request will be issued if the handshake is not bypassed by programing the CLPCR register bits 23–16. If the corresponding bits are set, then the request signal will not be issued to the corresponding module and CCM will not wait for its acknowledge in the process of entering low power mode. The requests will commence via the following signals:
   - SAHARA: sahara_ipg_stop_req
   - RTIC: rtic_ipg_stop_req
   - IPU: ipu_stop_clk_at_stop_req
   - SDMA: sdma_ipg_stop_req
   - SCC: scc_ipg_stop_req
   - EMI: emi_lpmd, emi_lpmd_fast, emi_lpmd_int1, emi_lpmd_slow, emi_lpmd_garb

NOTE

Since EMI needs the aclk_fast (ddr clock) to be running for it to answer with emi_lpmd, then software should make sure to turn on the aclk_fast (ddr_clk) before entering any low power mode that requests the emi_lpmd.

— AHBMAX: ahbmax_halt_req

Once the above modules finished their operation and are ready to enter low power mode, they acknowledge CCM that its safe to turn of their clocks. The acknowledge signals are as follows:

- SAHARA: sahara_ipg_stop_ack
- RTIC: rtic_ipg_stop_ack
- IPU: ipu_stop_clk_ack
- SDMA: sdma_ipg_stop_ack
- SCC: scc_ipg_stop_ack
- EMI: emi_lpack, emi_lpack_fast, emi_lpack_int1, emi_lpack_slow, emi_lpack_garb
- AHBMAX: ahbmax_halt_ack

The requests will be generated by CCM in stages. CCM will continue from stage to stage once all the acknowledges of a present stages were received. The stages are the following:

- Stage 1: SAHARA, RTIC, IPU, and SDMA
- Stage 2: SCC
- Stage 3: AHBMAX
- Stage 4: EMI

Once CCM receives all the acknowledge signals needed, and if at least 8 ipg_clk cycles has elapsed since the assertion of ccm_ipg_stop signal, then it will enter STOP mode and do the following tasks in the order they are written below:

1. * Close the system modules clocks, not including pgc_clk. The closing of clocks will include the ckil_sync clocks of Megamix only if the indication from GPC is that Megamix is about to be powered down (by spare_input_10 indication). The gating signal output for the CKIL sync gating cells is spare_output_0).
2. Assert system_in_stop_mode signal. This signal is connected to the IO pads for observability, to indicate of a STOP mode.

Once CCM is in STOP mode, it will check if TZIC_DSM_WAKEUP signal has asserted or if ARM_DSM_REQUEST has negated during the process of entering STOP mode. If it has occurred, then CCM will exit STOP mode.

If TZIC_DSM_WAKEUP is not asserted and ARM_DSM_REQUEST is asserted, then CCM will move to state STOP_GPC and generate a request to GPC to power down the ARM platform, IPU, VPU, GPU, GPU2D and EMI by asserting signals ccm_pdn_4arm_req and ccm_pdn_4all_req. If in GPC those modules are defined to be powered down on STOP mode then GPC will commence powering down for them.

After the GPC sends the pdn_ack signal, the following occurs:

1. Close PLLs and DPLL_ips by deasserting dpll_en_dpllip and ref_clk_en_dpllip.
2. After deassertion of lock ready flag from the PLL's, close CAMPs and FPM by deasserting ccm_CAMP1_en, ccm_CAMP2_en and ccm_fpm_en.
3. If sbyos bit was set, then de assert ref_en_b signal to close external oscillator and assert cosc_pwrdown to put on-chip oscillator in power down mode. (if they were not already closed by software. If one of them was already closed by software then discard the sbyos operation for it and perform the sbyos operation only on the one that is working. If both of them were closed in run mode, then discard sbyos operation for both of them).

   If vstby bit was set, then assert pmic_vstby_req signal, to indicate power management IC to shift voltage to standby voltage.

CCM's low power state machine will remain in state STOP_GPC until STOP mode will be exited.

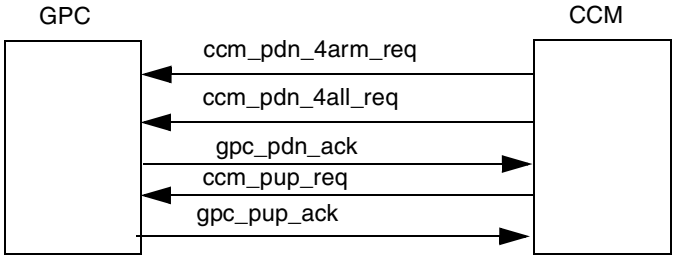


Figure 7-55. CCM-GPC Connectivity

7.4.8.3.2 Exiting Stop Mode

As soon as the unsynchronized TZIC_DSM_WAKEUP signal is seen as asserted or ARM_DSM_REQUEST is seen deasserted, the CCM exits STOP mode according to the following procedure

1. If vstby bit was set, deassert pmic_vstby_req to notify power management IC to change voltage from standby voltage to functional voltage.

   If sbyos was set, and CCM closed either external oscillator or on-chip oscillator, then CCM will start external oscillator and on-chip oscillator by asserting ref_en_b signal and deasserting cosc_pwrdown signal respectively.
2. After amount of CKIL's defined in stby_count bits, wait until pmic_vfunctional_ready signal is asserted. pmic_vfunctional_ready is used if an internal counter is not used. This is the notification from power management IC that the voltage is ready at its functional value. (Figure 7-58 describes the pmic signals connectivity.) Only then will CCM continue with the next steps:
3. Start FPM or CAMPs (based on definition of CAMP1_EN, CAMP2_EN and FPM_EN bits)
4. If any CAMP or on-chip oscillator were started, wait until oscnt has finished its counting to make sure that external oscillator, CAMPs output and on-chip oscillator are ready.
5. If FPM was selected, wait until assertion of fpm_lrf.
6. Start PLLs. Only the PLLs that were configured to be on prior to the entrance to STOP mode will be started.

7. Enable VPU, GPU2D, and GPU clocks only.

CCM will request GPC to restore power by GPC_PUP_REQ. If power was removed from the ARM platform or IPU or VPU or GPU2D or GPU, GPC will notify CCM by asserting signal GPC_PUP_ACK that power to ARM, VPU, IPU, GPU2D, GPU and EMI is back on, and its safe to exit from STOP mode. Only then CCM will do the following:

1. Once assertion of notification from SRC that the resets for the power gated modules has been finished, (src_power_gating_reset_done is set) negate the low power request signals to all modules and enable all modules clocks including ARM clocks and CKIL sync of Megamix (if it was turned off), and return to run mode. (the clocks that their CCGR bits define not to be open in RUN mode will not be opened, and will continued to be gated in RUN mode).
2. Once system is in run mode, negate signals ccm_ipg_stop and system_in_stop_mode.

Once the interrupt propagates through all the synchronization logic, the ARM CPU will recognize and service it. This will force the negation of the ARM_DSM_REQUEST output. Prior to this point, ARM_DSM_REQUEST and TZIC_DSM_WAKEUP were simultaneously asserted. Since TZIC_DSM_WAKEUP has top priority, the system is able to wake up.

Before the ARM exits the ISR, it will clear the interrupt source. This will cause the TZIC_DSM_WAKEUP signal to be negated. At this point, the two low power control signals are negated, and the STOP sequence can be repeated at any time.

Figure 7-56 describes the connectivity of ARM, CCM and TZIC.

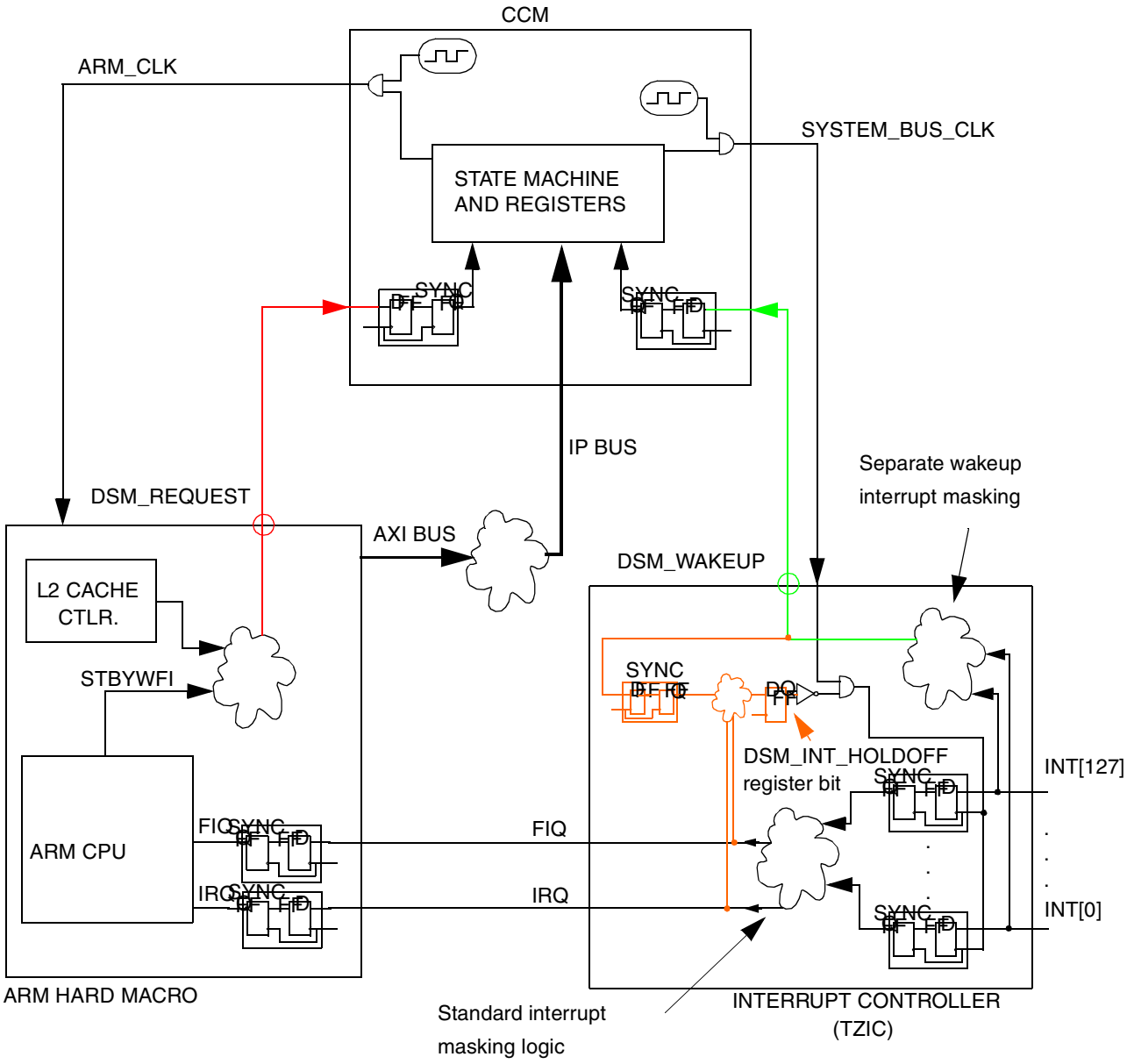


Figure 7-56. ARM Low Power Request

Figure 7-57 describes CCM's low-power state machine:

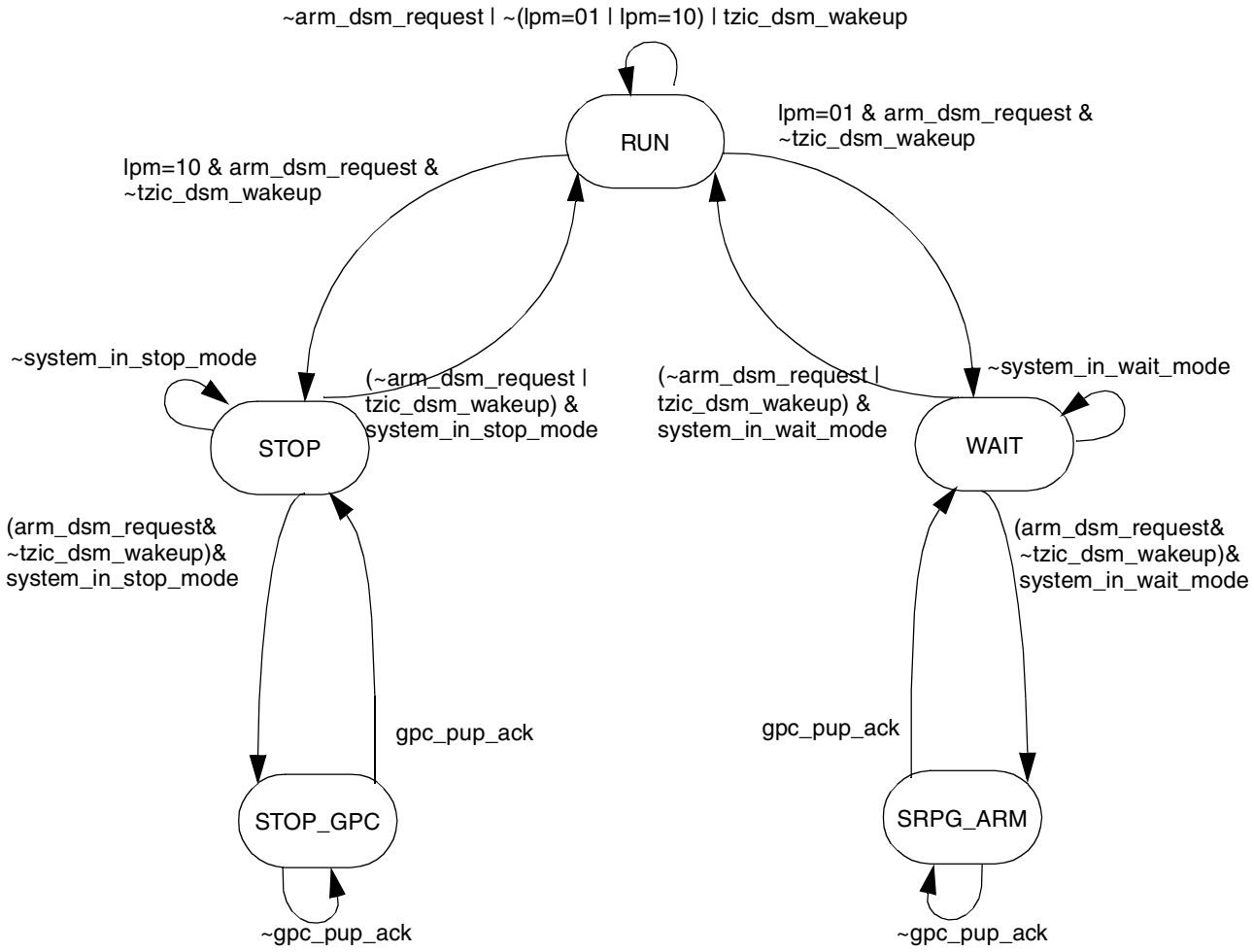


Figure 7-57. CCM Low Power State Machine

7.4.8.3.3 PMIC Signal Description

stby_count value should be larger than t1 - larger than the amount of time that it takes between CCM's negation of pmic_vstby_req, and the negation of pmic_vfuncional_ready (the signal coming back from PMIC to indicate that the voltage started to change).

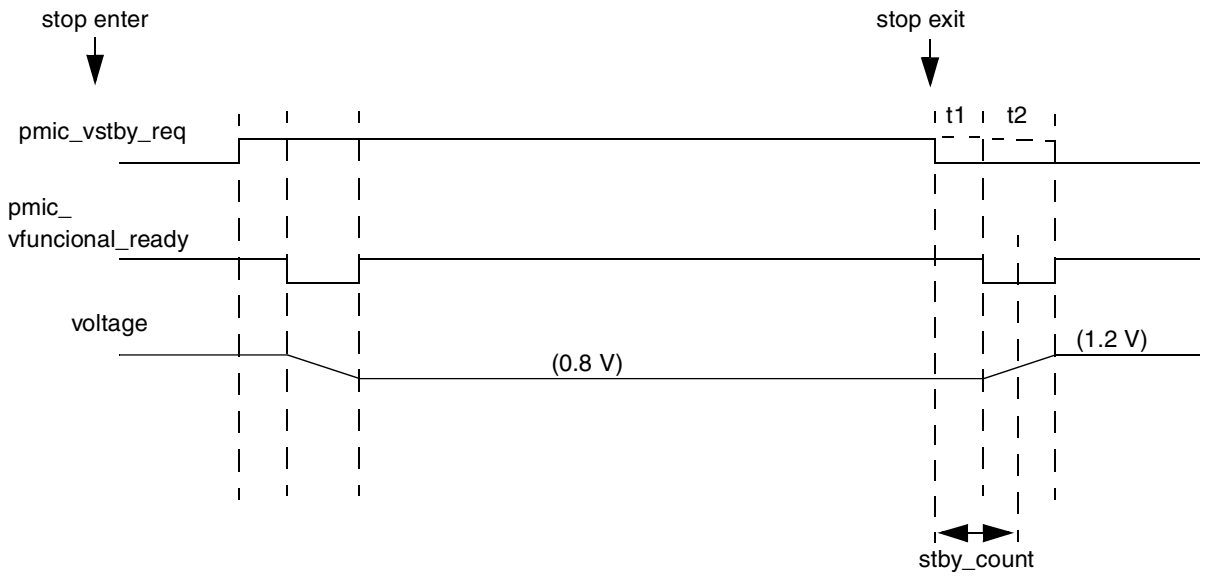


Figure 7-58. PMIC Signal Description

Note: Software should configure the IOMUXC so that pmic_vstby_req and pmic_vfuncional_ready will be operating in the correct ALT mode, so that they will be connected between CCM and the pads.

7.4.8.4 Low Power Screen Refresh Mode (LPSR)

The following section describes the solution for the low-power screen refresh support for the i.MX51. This allows the IPU to refresh the screen by reading the data from the internal RAM (incorporated in SCC module), while the system is in low-power mode by running from a low frequency clock. The low frequency clock source can be chosen from three possible options. LPSR is similar to STOP mode with the distinction that IPU can continue to access internal memory. For this capability, the clock to IPU, internal memory system, and emi master 0 should continue to function. All other clocks will be closed, including PLLs. When entering this mode, it is not allowed to configure IPU and emi to power gating / SRPG.

To be able to supply the above clocks, ccm generates lpsr_clk_root. This clock is generated from the following options:

- FPM clock output
- FPM clock output divided by 2
- GND. To reserve power, this option is to be chosen when LPSR is not used.

The lpsr_clk_root is connected to LPCG. The above clocks that need to continue to function in LPSR will have a mux in LPCG to shift between their functional clock to the lpsr_clk_root. The control of the mux comes from ccm and is named LPSR_mode_clk_en.When this signal is ‘0’ the mux will choose the functional clock for the above modules. When this signal is ‘1’ the mux will choose the lpsr_clk_root as the fed clock for the above modules. Figure 7-59 describes the LPCG part implemented for those clocks:

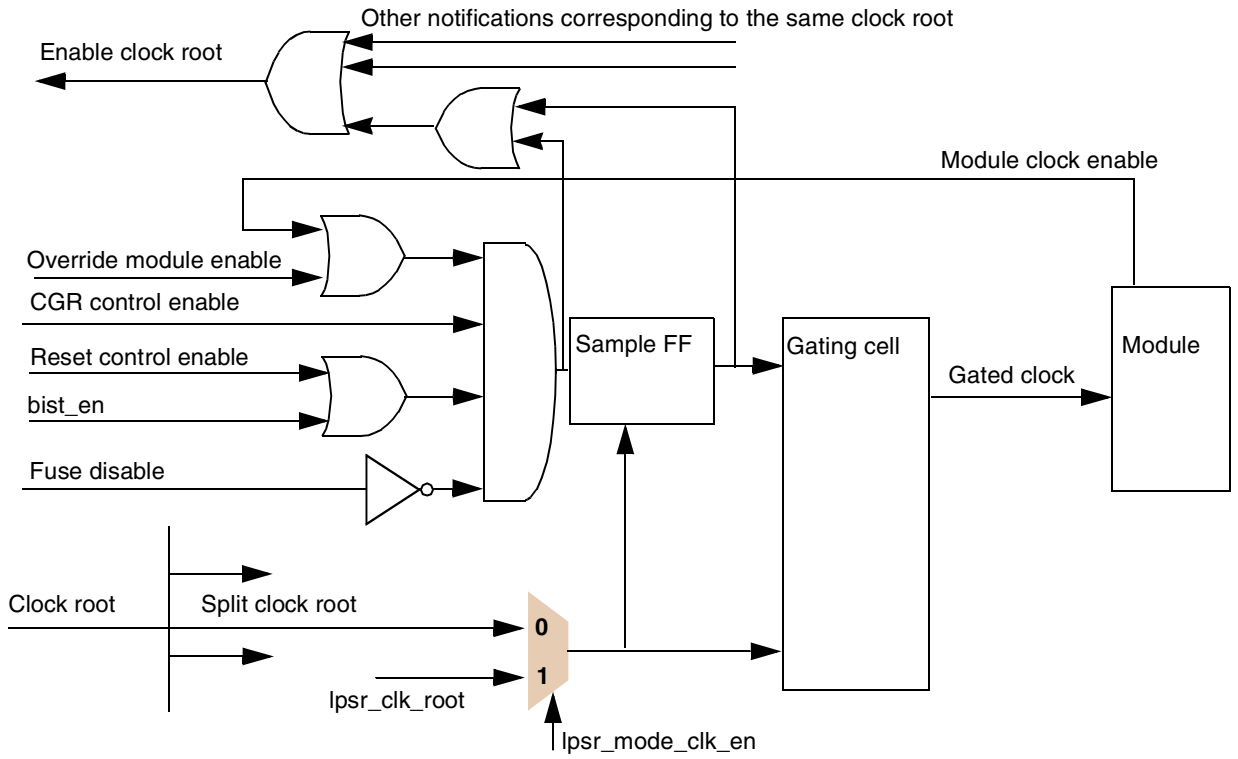


Figure 7-59. LPCG Part Implementation

Note: the IPU module expects specific configuration prior entering LPSR mode—please refer to IPU chapter for details.

7.4.8.4.1 LPSR Mode

LPSR mode uses the following procedure:

1. ARM will write to LPM bit to set it for LPSR mode.
2. ARM will write to DSM_INT_HOLDOFF bit in TZIC.
3. DSM_INT_HOLDOFF assertion forces the interrupt controller to stop the synchronizer clock for incoming interrupts. Any interrupts that had previously been synchronized by the interrupt controller will remain asserted.
4. If a wakeup event occurred or an interrupt was serviced or is pending, the CPU may abort the DSM sequence by exiting the main shutdown code sequence and returning to the desired security mode.

   If no wakeup event has occurred the software should execute the Wait For Interrupt (WFI) instruction.
5. Upon execution, the WFI instruction will cause the ARM to drain its write buffers and enter a quiescent state. At this point the ARM will assert the STANDBYWFI signal.

6. The ARM platform will then wait until the L2 cache controller has become inactive. Once the L2 is inactive and the CPU's STANDBYWFI output is asserted, the ARM platform will assert the ARM_DSM_REQUEST signal to the CCM.
7. The CCM will continuously synchronize both the ARM_DSM_REQUEST and TZIC_DSM_WAKEUP inputs into the clock domain used by its internal state machine.The only time when those signals will not be synchronized is, when PLLs will be closed. In this case the exiting from LPSR mode will be done asynchronously (please refer to the section below describing the exiting from STOP mode).
8. If TZIC_DSM_WAKEUP is negated and ARM_DSM_REQUEST is asserted, the CCM will begin the LPSR shutdown sequence:
   a) CCM will prepare the LPSR root clock - if FPM was chosen, it will start FPM and wait until FPM_lrf is asserted.
   b) CCM will stop ARM clock. ARM clock can be stopped even before FPM_lrf is asserted.
   c) CCM will assert ccm_ipg_stop and ccm_int_mem_ipg_stop to indicate that it has started the STOP entrance procedure.
9. CCM will request an acknowledge to close clocks of SAHARA, RTIC, IPU, SDMA, SCC, EMI and AHBMAX if their clocks was not already closed in run mode. The requests will commence via the following signals:
   — SAHARA: sahara_ipg_stop_req
   — RTIC: rtic_ipg_stop_req
   — IPU: ipu_stop_clk_at_stop_req
   — SDMA: sdma_ipg_stop_req
   — SCC : scc_ipg_stop_req
   — EMI: emi_lpmd, emi_lpmd_fast, emi_lpmd_int1, emi_lpmd_slow, emi_lpmd_garb

   NOTE

   Since EMI needs the aclk_fast (ddr clock) to be running for it to answer with emi_lpmd, software should make sure to turn on the aclk_fast (ddr_clk) before entering any low power mode that requests the emi_lpmd.

   — AHBMAX: ahbmax_halt_req
10. Once the above modules finished their operation and are ready to enter low power mode, they acknowledge CCM that its safe to turn of their clocks. The acknowledge signals are as follows:
   — SAHARA: sahara_ipg_stop_ack
   — RTIC: rtic_ipg_stop_ack
   — IPU: ipu_stop_clk_ack
   — SDMA: sdma_ipg_stop_ack
   — SCC: scc_ipg_stop_ack
   — EMI: emi_lpack, emi_lpack_fast, emi_lpack_int1, emi_lpack_slow, emi_lpmd_garb
   — AHBMAX: ahbmax_halt_ack

11. The requests will be generated by CCM in stages. CCM will continue from stage to stage once all the acknowledges of a present stages were received. The stages are as follows:
    - Stage 1: SAHARA, RTIC, IPU, SDMA
    - Stage 2: SCC
    - Stage 3: AHBMAX
    - Stage 4: EMI

Once CCM receives all the acknowledge signals needed, and if at least 8 ipg_clk cycles has elapsed since the assertion of ccm_ipg_stop signal, then it will enter STOP mode and does the following tasks in the order they are written below:

1. Close the system modules clocks, not including pgc_clk.
2. Assert system_in_stop_mode signal.

Once CCM is in STOP mode, it will check if TZIC_DSM_WAKEUP signal has asserted or if ARM_DSM_REQUEST has negated during the process of entering STOP mode. If it has occurred, then CCM will exit STOP mode.

If TZIC_DSM_WAKEUP is not asserted and ARM_DSM_REQUEST is asserted, then CCM will move to state STOP_GPC and generate a request to GPC to power down the ARM platform, GPU, GPU2D and VPU by asserting signals ccm_pdn_4arm_req and ccm_pdn_4all_req. If in GPC those modules are defined to be powered down on STOP mode then GPC will commence powering down for them.

NOTE

IPU is not allowed to be powered down in LPSR mode. When finished, the GPC sends the pdn_ack signal.

In parallel to the request to GPC to power down the needed modules, the following takes place:

1. Assert lpsr_mode_clock_en to shift the lpcg mux to the lpsr_clk_root.
2. Open the clocks of the modules involved in LPSR, that is, IPU and emi aclk_m3 (ipu_clk_enable), emi (emi_int1_clk_enable), downsizer, axi2ahb, regslice, ahbmux (ahbmux2_clk_enable), scc (scc_clk_enable).
3. Deassert ccm_int_mem_ipg_stop and scc_ipg_stop_req (connected to scc to notify it to exit STOP mode).
4. Deassert emi_lpmd, emi_lpmd_int1 (connected to emi to notify it to exit low power mode on int1 and on master side).
5. Assert ccm_lpsr_ipu to notify IPU to enter LPSR mode.

After the GPC sends the pdn_ack signal, the following takes place:

1. Close PLLs and dpll_ips by deasserting dpll_en_dpllip and ref_clk_en_dpllip.
2. Close CAMPs and FPM (only if FPM is not the source for lpsr_clk_root) by de asserting ccm_CAMP1_en, ccm_CAMP2_en and ccm_fpm_en.

3. If sbyos bit was set, then de assert ref_en_b signal to close external oscillator and assert cosc_pwrdown to put on-chip oscillator in power down mode. (if they were not already closed by software. If one of them was already closed by software then discard the sbyos operation for it and perform the sbyos operation only on the one that is working. If both of them were closed in run mode, then discard sbyos operation for both of them).
4. If vstby bit was set, then assert pmic_vstby_req signal, to indicate power management IC to shift voltage to standby voltage.

CCM's low power state machine will remain in state "STOP_GPC" until LPSR mode is exited.

7.4.8.4.2 Exiting LPSR Mode

As soon as the unsynchronized TZIC_DSM_WAKEUP signal is seen as asserted or the ARM_DSM_REQUEST is negated, the CCM begins the process of exiting LPSR mode, according to the following procedure:

1. If vstby bit was set, deassert pmic_vstby_req to notify power management IC to change voltage from standby voltage to functional voltage.

   If sbyos was set and CCM closed either external oscillator or on-chip oscillator, the CCM will start external oscillator depending on the configuration of the on-chip oscillator by asserting ref_en_b signal and deasserting cosc_pwrdown signal respectively.
2. After amount of CKILs defined in stby_count bits, wait until pmic_vfunctional_ready signal is asserted. This is the notification from power management IC that the voltage is ready at its functional value. (Figure 7-58 describes the pmic signals connectivity.) Only then will CCM continue with the next steps.
3. Start FPM (if not already working) or CAMPs (based on definition of CAMP1_EN, CAMP2_EN and FPM_EN bits).
4. If any CAMP or on-chip oscillator were started, wait until oscnt has finished its counting to make sure that external oscillator, CAMPs output and on-chip oscillator are ready.
5. If FPM was selected, wait until assertion of fpm_lrf (if its not already working).
6. Start PLLs. Only the PLLs that were configured to be on prior to the entrance to LPSR mode will be started.
7. Enable VPU, GPU2d and GPU clocks only.

CCM requests that GPC restores power by GPC_PUP_REQ. If power was removed from the ARM platform or IPU, VPU, GPU2D, GPU, GPC notifies CCM by asserting signal GPC_PUP_ACK that power to ARM, IPU, VPU, GPU2D, GPU and EMI is back on, and it is safe to exit from LPSR mode. Only then does CCM do the following:

1. Once assertion of notification from src that the resets for the power gated modules has been finished, (src_power_gating_reset_done is set) deassert ccm_lpsr_ipu.
2. Once ipu_lpsr_wakeup_ack is asserted, assert emi_lpmd, emi_lpmd_int1 (connected to EMI to notify it to enter low power mode on int1 and on master side), and ccm_int_mem_ipg_stop and scc_ipg_stop_req (connected to scc to notify it to enter STOP mode).

3. Once emi_lpack_int1 and scc_ipg_stop_ack are received, close the clocks of the modules involved in LPSR, that is, IPU and emi aclk_m3 (ipu_clk_enable), emi (emi_int1_clk_enable), downsizer, axi2ahb, regslice, ahbmux (ahbmux2_clk_enable), scc (scc_clk_enable). emi_lpack is ignored since it willl not be received - the fast clock is not available in this state, hence emi will not be able to generate the emi_lpack. The emi_lpack_int is an enough indication that emi has finished its function, since IPU has already stopped.
4. Deassert lpsr_mode_clock_en to shift the lpcg mux from the lpsr_clk_root to the functional root clock.
5. Enable all modules clocks including ARM clocks and return to run mode.
6. Once system is in run mode, negate signals ccm_ipg_stop and system_in_stop_mode and all low power requests (emi_lpmd, emi_lpmd_int1, emi_lpmd_fast, emi_lpmd_slow, scc_ipg_stop_req, and so on.).

Once the interrupt propagates through all the synchronization logic, the ARM CPU will recognize and service it. This will force the negation of the ARM_DSM_REQUEST output. Prior to this point, ARM_DSM_REQUEST and TZIC_DSM_WAKEUP were simultaneously asserted. Since TZIC_DSM_WAKEUP has top priority, the system is able to wake up.

Before the ARM exits the ISR, it will clear the interrupt source. This will cause the TZIC_DSM_WAKEUP signal to be negated. At this point, the two low power control signals are negated, and the STOP sequence can be repeated at any time

7.4.8.5 LPMD Request from SRC (Reset Controller)

SRC generates a signal named emi_DVFS_req. This signal will be asserted in case of WARM reset to request emi to enter sdram into selfrefresh. Since CCM is also a generator for the request to emi for entering sdram to selfrefresh, then the SRC request will be connected to CCM, and CCM will generate one common request to emi for sdram selfrefresh. This common request will assert either if CCM is the originator of the request or if the SRC is the originator of the request. The SRC request (named emi_DVFS_req) is connected to CCM's input src_warm_DVFS_req. CCM generates the request to EMI via DVFS request signal. This signal generation will be combined with the signal coming from SRC (src_warm_DVFS_req) to generate one DVFS request signal to EMI.

The acknowledge from emi regarding the self-refresh of EMI is connected to both CCM and SRC.

7.4.8.6 Low Power Audio Playback Mode (LP-APM)

Low Power Audio Playback mode (APM) defines special low power mode, dedicated for Audio only playback mode. It involves PLL on/off and frequency settings, as well as voltage and clock gating aspects, to enable lowest possible power consumption.

This section covers APM, as well as assumptions, and enter/exit flows from the APM mode.

7.4.8.6.1 Low Power APM Mode Definition

APM can be entered by DVFS settings or by manual clock configuration.

Only one PLL of i.MX51 is running—ARM PLL. This PLL is set to the maximal frequency that ARM can run at that lowest point, while still ensuring ARM and bus system processing needs. .

The peripherals will also work on low voltage. The peripherals work on a frequency suitable for this low voltage (about third of the nominal frequency). For information regarding voltages at low performance operating mode, please refer to the i.MX51 Datasheet.

For i.MX51 used as master on SSI clocks, the MCU PLL frequency must be set to an integer multiplication of SSI clock

7.4.8.6.2 LP-APM Mode Restrictions

The following restrictions apply to LP-APM mode restrictions.

- Enter and exist to/from Low Power APM mode has effect on Display and SSI clocks. If i.MX51 is master on SSI clocks, audio playback will be interrupted during the switch, and as such - it is not recommended to switch modes during audio playback.
- Due to impact to Display controller clock, switch to/from LP-APM, SW should take that into consideration. This is not applicable when using Smart Display.

Figure 7-60 describes the sequence to be performed to switch from a high power settings (3 PLLs, high voltage) to the low power APM.

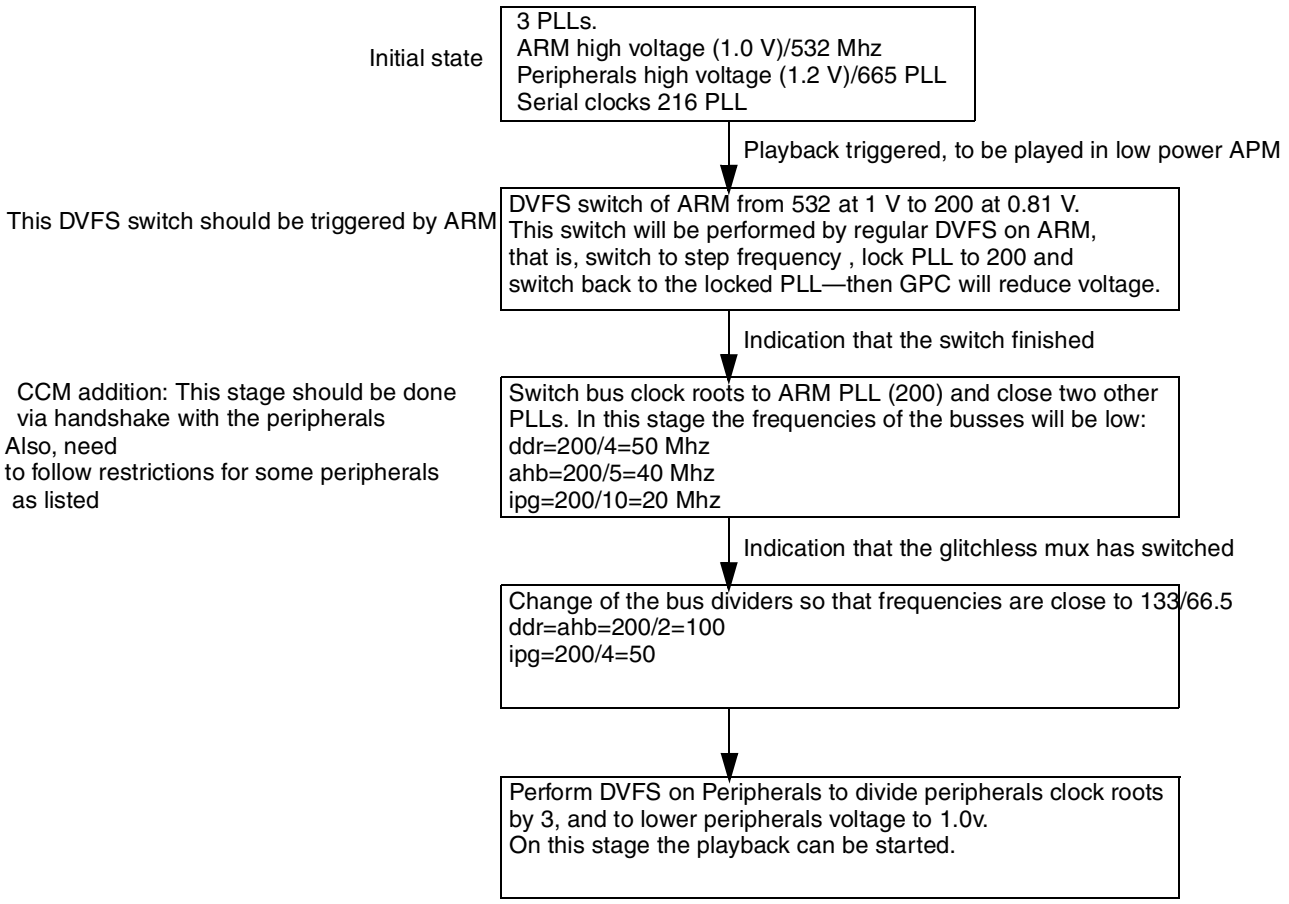


Figure 7-60. Flow of Entering Low Power APM on i.MX51

Table 7-44 shows the LP-APM flow frequencies.

Table 7-44. LP-APM Flow Frequencies

Stage	Farm [Mhz]	Varm [v]	Fahb [Mhz]	Fipg [Mhz[	Vper [v]	Comments
Initial state	532	1.0	133	66.5	1.2	run system from 3 PLLs: ARM PLL = 532 Mhz supplying ARM PERIPH PLL = 665 Mhz supplying busses (ahb, ipg). Serial PLL = 216 Mhz supplying USB, TVE, and so on.
DVFS on ARM	200	0.81	133	66.5	1.2	GPC operation triggered by software
Switch bus clocks to ARM PLL	200	0.81	40	20	1.2	Peripheral dividers are not changed during the shift. The shift should involve EMI handshake
Change bus dividers	200	0.81	100	50	1.2	—

Table 7-44. LP-APM Flow Frequencies

Stage	Farm [Mhz]	Varm [v]	Fahb [Mhz]	Fipg [Mhz[	Vper [v]	Comments
DVFS on peripherals	200	0.81	50	25	1.0	DVFS on peripherals with div=2
DVFS on peripherals	200	0.81	33.3	16.66	1.0	DVFS on peripherals with div=3

Figure 7-62 shows the frequency and voltage flow

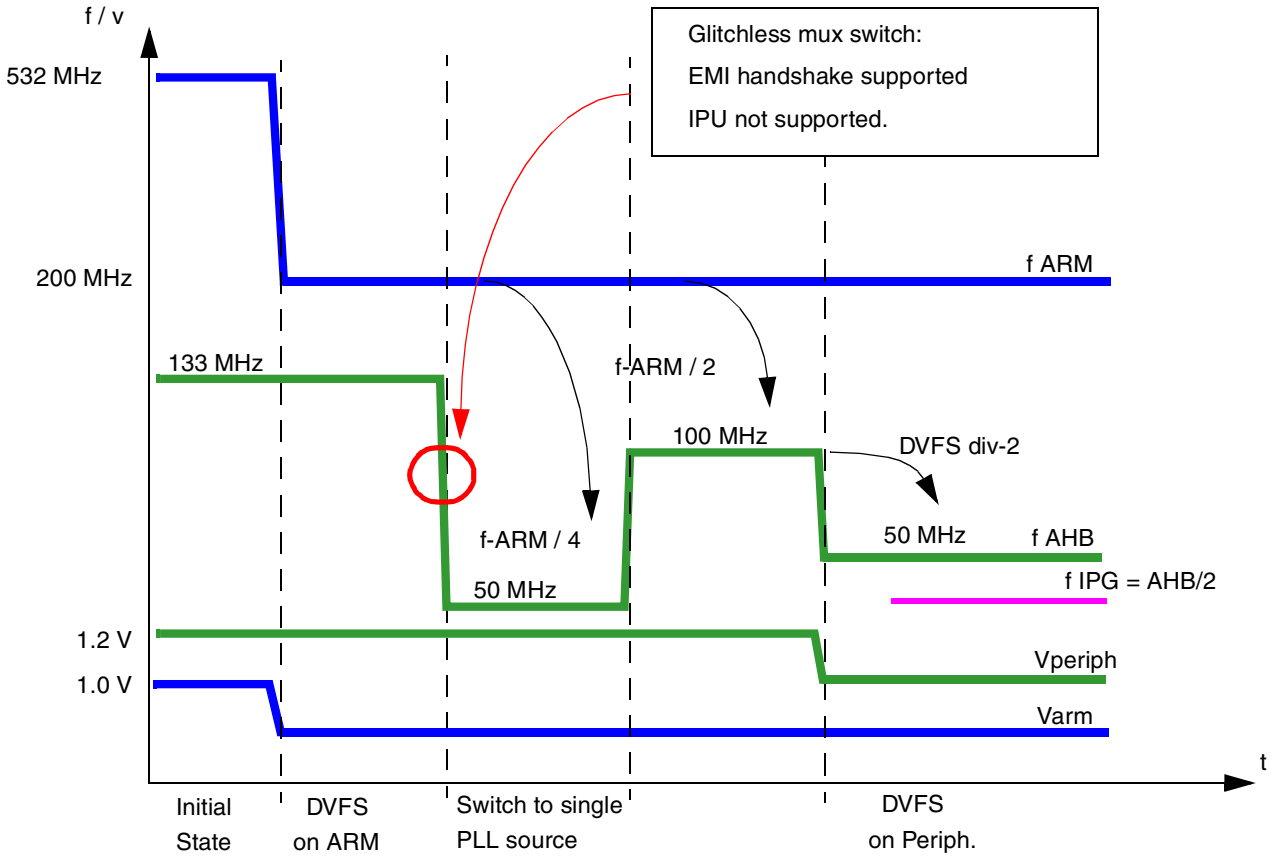
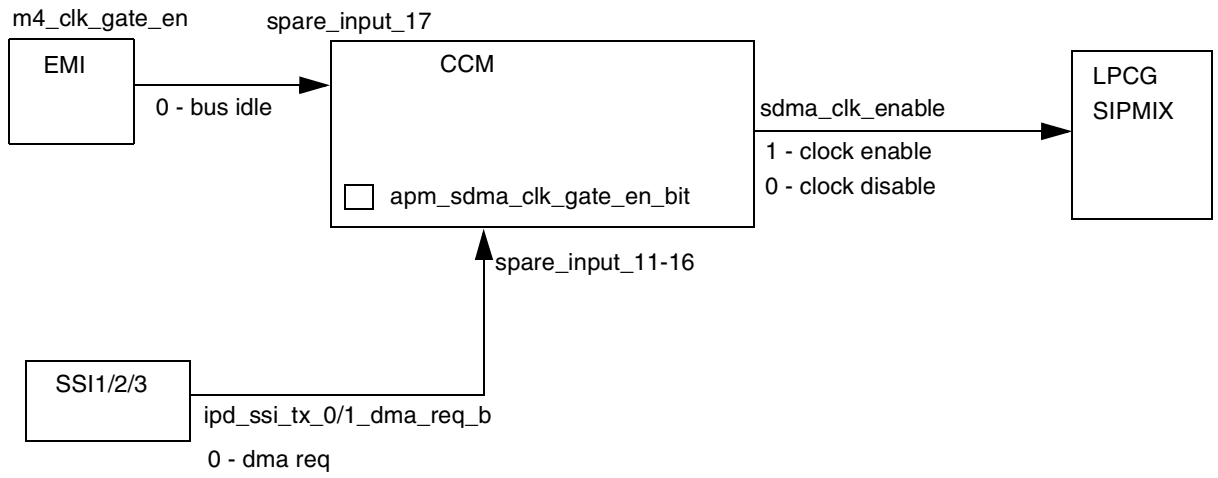


Figure 7-61. Frequency and Voltage Flow

7.4.8.7 SDMA Clock Gating in APM Scenario

In LP-APM scenario, SDMA accesses EMI through master 4. Once EMI notifies that master 4 is idle (based on auto clock gating of EMI), then we can assume that all the data needed was transferred to SDMA, and SDMa clocks can be gated off. The SDMA clocks should be turned on, once SSI requests the next data from SDMA. Between those two indications, SDMA clocks can be gated off, by using sdma_clk_enable output of CCM. All this operation will be masked by apm_sdma_clk_gate_en_bit. If this

bit is not enabled, then the SDMA clock gating operation will not be based on m4_clk_gate signal. Figure 7-62 describes the design implementation.



CCM logic:
EMI bus idle → sdma_clk_enable = 0 (disable SDMA clock).
Upon SSI request (any of ipd_ssi_* = 0) → sdma_clk_enable = 1 (enable SDMA clock).
Wait for EMI bus idle to assert (m4_clk_gate_en = 1) → start from begining to look for bus idle.
The above should be masekd if the apm_sdma_clk_gate_en_bit is deasserted, (sdma clk enable).

Figure 7-62. Design Implementation Scheme

7.4.8.8 Wake Up Detector

When the MEGAMIX is in SRPG mode, all the modules in the MEGAMIX do not receive clock and power; hence they cannot wake up the system from external interrupts. The Wake Up Detector allows the reception of the following external interrupts when the MEGAMIX is SRPGed:

- KPP
- SDHC 1/2 Detection
- GPIO1_4, GPIO1_5, GPIO1_6, GPIO1_7, GPIO1_8, GPIO1_9

The Wake Up detector catches interrupts only when the clocks are stopped, that is, upon dpll_en_dpllip signal deassertion. In order to receive an interrupt, a corresponding pin has to be configured at IOMUX controller as described below.

7.4.8.8.1 KPP Interrupt

Interrupt occurs when key pressed. To enable the interrupt set corresponding bit at CCM_CIMR register to zero.

7.4.8.8.2 SDHC Interrupts

Interrupt occurs when SD card is inserted. To enable the interrupt set corresponding bit at CCM_CIMR register to zero. Each SD has separate enable and status bit.

7.4.8.8.3 GPIO Interrupts

The i.MX51 can wake up from STOP mode with MEGAMIX at SRPG state by one of the six GPIO interrupt described above.

Each interrupt can be masked at CCM_CIMR register. GPIO direction and interrupt type have to be configured for each GPIO at CCM_CR2 register, as it is configured at GPIO module.

7.4.8.9 Recommendations for Using Low Power Consumption from CCM

In addition to using APM mode for low power, here are the recommendations for configuring CCM so that it consumes less power. These recommendations should be followed when applicable:

- Use the AHB clock source as the source of all AXI buses and VPU core clkin - this case the power of the AXI dividers will be preserved.
- Work from one PLL and disable the other two PLLs.
- Generate SSI external clocks from SSI clock roots—in this case the SSI external clocks dividers power will be preserved.
- Generate SPDIF clocks from SSI clock roots—in this case the SPDIF clocks dividers power will be preserved. Gate debug clocks (arm debug clock, CTIs, DAP,...) and functional clocks that are not required for operation by CCGR registers. If a whole root clock is not required and gated by CCGR bits, it will gate the clock entering the divider.
- Shut external oscillator when not required.
- Lower the voltage and frequency (DVFS) when possible.
- Enable dynamic clock gating (enables from modules) so the clocks will be gated automatically when not used.
- Use the lowest possible frequencies of the PLLs to generate low frequencies that will enter the clock dividers and as a result save power consumed by the CCM.

Chapter 8
Debug Architecture

8.1 Overview

This chapter describes the debug architecture.

8.1.1 Introduction

This chapter describes the hardware and software debug and application development features and resources of i.MX51. There are core/platform-specific resources, resources associated with some of the more complex IP blocks, and chip-wide resources. Also discussed is the interface to external debug and development tools.

The debug architecture of i.MX51 is based on that of previous members of the i.MX application processor family.

An overview of the primary debug capabilities and features includes:

- JTAG-based access (control, monitoring), built around the Secure Controller (SJC)
- Trace Port
- Real-time and Halt-mode debug and profiling capabilities of the Cortex-A8 core platform
- Real-time and Halt-mode debug capabilities of the SDMA core
- An SOC-wide cross-trigger system built around ARM's Embedded Cross Trigger (ECT)
- Visibility of pre-selected critical internal signals via pin muxing
- External memory interface/controller arbitration profiling

Each of these features will be described in detail in the following sections of this chapter. The i.MX51 also supports ARM CoreSight architecture for system debug and trace capabilities.

8.1.2 Debug Strategy

The following features form the building blocks of the i.MX51 Debug design:

- Software debug (MSFT kernel)
- Trace the CPU activity via ETM trace port or ETB (ARM only)
- Control and monitor via JTAG
- Monitor preselected critical internal signals via pin muxing - scope

8.2 System Secure Controller—SJC

The SJC module is the bridge between external development and test instrumentation and the internal JTAG-accessible debug and test resources. It implements and manages the daisy-chained topology consisting of it's own TAP and those of the SDMA, the ARM core, and the ARM Debug Access Port (DAP). The ARM core is not part of a daisy chain.

8.2.1 JTAG Topology

Figure 8-1 shows a high level block diagram of the i.MX51 JTAG connectivity.

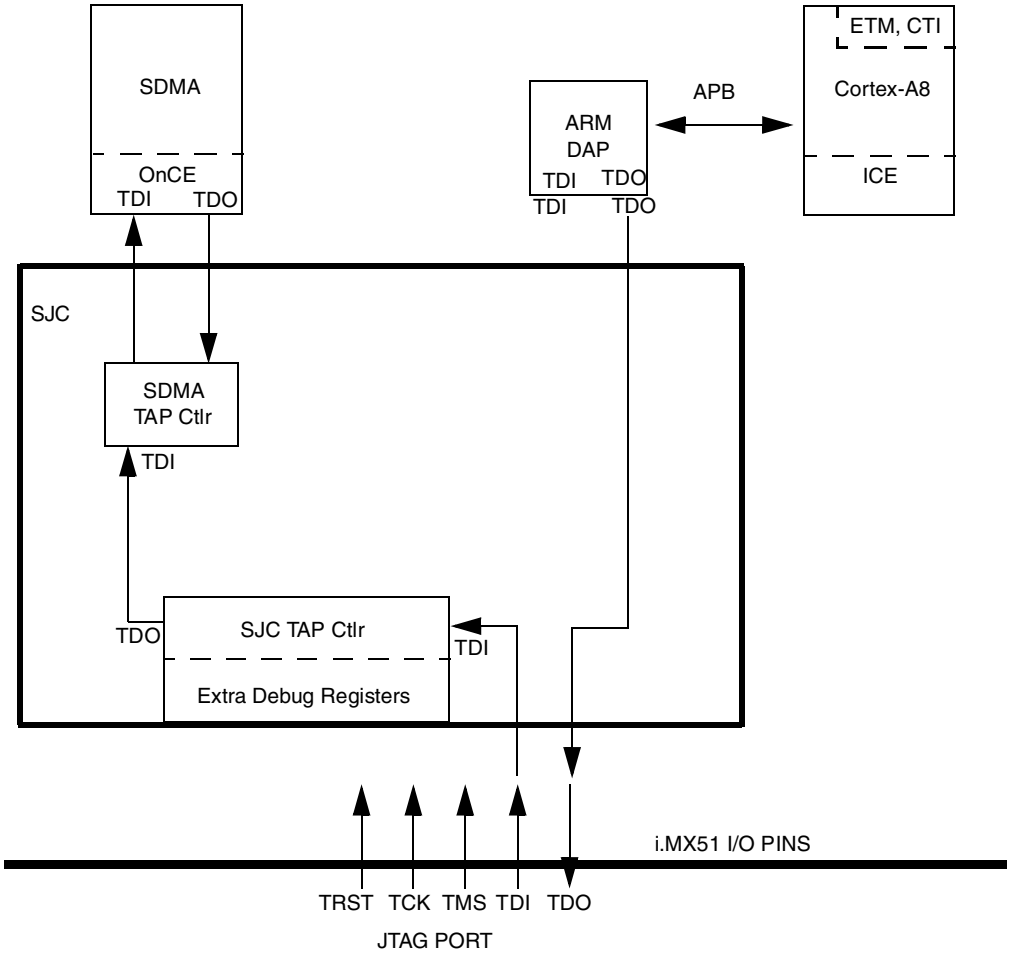


Figure 8-1. JTAG Connectivity

8.2.2 Secure JTAG Controller Main Feature

The secure JTAG controller main features are as follows:

- IEEE 1149.1 (standard JTAG) interface to off-chip test and development equipment

— Includes an SJC-only mode for true IEEE 1149.1 compliance, used primarily for board-level implementation of boundary scan.

- Debug-related control and status, such as putting selected cores into reset and/or debug mode and the ability to monitor individual core status signals via JTAG.
- System status, such as the state of the PLLs (locked or not locked)
- Generic status and control, defined on a per-SOC basis by the architecture team
- Factory test related control/status such as PLL bypass and memory BIST
- Four levels of security, ranging from no security to no JTAG accessibility to the chip.

8.2.3 SCJ TAP Port

The SJC in the i.MX51 supports the following standard JTAG pins: TRSTB, TDI, TDO, TCK, and TMS.

8.2.4 SJC Main Blocks

- Interface to the outside world via the standard JTAG pins
- Interface to the external Debug_Event pin
- A master TAP controller, which implements the standard JTAG state machine.
- Implementation of the mandatory and optional IEEE P1149.1 (JTAG) instructions
  — Mandatory: “EXTEST”, “SAMPLE/PRELOAD,” and “BYPASS”
  — Optional: “ID_CODE” (SOC JTAG ID register), “HIGHZ”
- Supports the SDMA’s DR-path-only JTAG architecture by implementing the controller portion of its TAP (including “BYPASS” as the default state) within the SJC
- Provides the serial interfaces to various DSP-domain JTAG resources (not implemented on i.MX51)
- The ExtraDebug registers, which implement a variety of control and status features
  — Three 32-bit insecure general purpose status registers
  — Two 32-bit secure status registers - one predefined, one general purpose.
  — Control and status registers for debug, core, charge pump, PLL, and BIST related functions
  — Control bits for memory timing control (not used on i.MX51)
- Four levels of fuse-defined security, ranging from no security to no access.

8.2.5 SJC Features—JTAG Disable Mode

In addition to four different JTAG security modes that are implemented internally in the Secure JTAG Controller (SJC), there is an option to disable the SJC functionality by e-FUSE configuration. This creates an additional JTAG mode, JTAG Disabled, with the highest level of JTAG protection. In this mode, all JTAG features are disabled. Specifically, the following debug features are disabled in addition to the features that were already disabled in “No Debug” JTAG mode:

- Memory BIST
- Boundary scan register (BSR)

- Non-Secure JTAG control registers (PLL configuration, Deterministic Reset, PLL bypass)
- Non-Secure JTAG status registers (Core status)
- Chip Identification Code (IDCODE)

8.2.6 SJC Registers Summary

The following is a summary of the SJC registers.

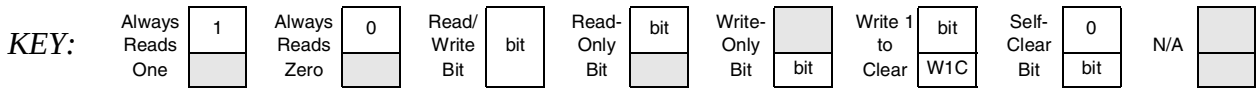


Table 8-1. SJC Registers Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPUSR1 ($BASE + 0x00)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	PLL3_lrf	PLL2_lrf	PLL1_lrf	0	0	S_STAT[1:0]			A_WFI	A_DBG
	W																
GPUSR2 ($BASE + 0x01)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	sdma_debug_core_state[3:0]			
	W																
GPUSR3 ($BASE + 0x02)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	ccm_ref_en_b	0	0	ccm_pdn_4all_req	ccm_pdn_4arm_req	system_in_stop_mode	system_in_wait_mode	ccm_ipg_stop	ccm_ipg_wait
	W																

Table 8-1. SJC Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSR ($BASE + 0x05)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	iim_fuse_latched
	W																
	R	0	~src_int_boot	ipt_secur_block	RSSTAT		sjm		ft	bsf	rsf	ebg	ebf	swe	swf	kta	ktf
	W																
GPCCR ($BASE + 0x07)	R W	jtag_sw_rst	dpdck0_2_en_pll3	pdf_pll3[3:1]			~pdf_pll3[0]	ref_clk_div_pll3	mfi_pll3[3:0]				dpdck0_2_en_pll2	pdf_pll2[3:1]			~pdf_pll2[0]
	R W	ref_clk_div_pll2	mfi_pll2[3:0]				dpdck0_2_en_pll1	pdf_pll1[3:1]			~pdf_pll1[0]	ref_clk_div_pll1	mfi_pll1[3:0]				0
PLLBR ($BASE + 0x08)	R W	sjc_ac_scan_cpd_pll3[3:0]			sjc_ac_scan_cpd_pll2[3:0]				sjc_ac_scan_cpd_pll1[3:0]					0	pll_bypass_en2	pll_bypass_en1	pll2_bypass_alternative
	R W	pll1_bypass_alternative	sjc_crstrt_pll3	sjc_crstrt_pll2	sjc_crstrt_pll1	sjc_cpres_pll3	sjc_cpres_pll2	sjc_cpres_pll1	sjc_cpen_pll3	sjc_cpen_pll2	sjc_cpen_pll1	sjc_control_sel_pll3	sjc_control_sel_pll2	sjc_control_sel_pll1	pll_bypass_en3	PLLBYPASS	PLLBPEN
GPUCR1 ($BASE + 0x09)	R	efuse_prog_supply_gate	0	0	0	0	ipt_sjc_test_mode[1:0]		ipt_sjc_lch_mode[2:0]			0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	ipt_bipg_tdien	0	0	0	0	usbphy_source_sel	0
	W																

Table 8-1. SJC Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPUCR2 ($BASE + 0x0A)	R	0	0	init_ssi_apm_clk_sel		init_periph_clk_sel	~init_periph_apm_sel[1]	init_periph_apm_sel[0]	~init_nfc_podf[2]	init_nfc_podf[1:0]		~init_emi_slow_podf[2]	init_emi_slow_podf[1:0]		~init_axi_b_podf[2]	init_axi_b_podf[1]	~init_axi_b_podf[0]
	W																
	R	init_axi_a_podf[2]	~init_axi_a_podf[1:0]		~init_ahb_podf[2]	init_ahb_podf[1:0]		init_ipg_podf[1]	~init_ipg_podf[0]	~init_perclk_podf[2]	init_perclk_podf[1]	~init_perclk_podf[0]	init_perclk_pred2[2:0]			init_perclk_pred1[1]	~init_perclk_pred1[0]
	W																
GPUCR3 ($BASE + 0x0B)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	sjc_ipt_io_tail_bypass	sjc_interrupt
	W																
	R	arm_clk_off_on_lpm	owire_line_alt7_hard_en	en_tester_cntl	smart_bypass	0	sjc_sre	sjc_sr_fast	sjc_sr_slow	sjc_dse_en	sjc_dse1	sjc_dse0	ahb_clk_bypass	emi_slow_clk_bypass	axi_a_clk_bypass	0	0
	W																
BISTCR1 ($BASE + 0x0F)	R	0	0	0	sdma_rom_gbist_addr[9:0]										0	0	gbist_rowfast
	W																
	R	gbist_mode[2:0]			gbist_reset	0	0	0	0	0	0	0	0	mcu_gbist_select	sdma_gbist_select	invoke	release_en
	W																
BISTCR2 ($BASE + 0x10)	R	0	0	0	0	0	0	mcu_rom_gbist_addr[13:4]									
	W																
	R	mcu_rom_gbist_addr[3:0]				0	0	0	0	0	0	0	0	0	0	0	0
	W																

Table 8-1. SJC Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BISTCR4 ($BASE + 0x12)	R	bist_en_wrck		gpu2d_bist_clk_en	bist_en_sms_clk_tve	bist_en_rom_36k_gbist	~L2Data_bist_clkdiv[2]	L2Data_bist_clkdiv[1:0]		L2Cache_bist_clkdiv[2]	~L2Cache_bist_clkdiv[1:0]		ETB_bist_clkdiv[2:1]		~ETB_bist_clkdiv[0]	VPU_bist_en_sms_clk	tigerp_vl_bist_en
	W																
	R	megamix_bist_en_sms_clk	IPU_bist_en_sms_clk	0	GPU_bist_en_sms_clk	EMI_bist_en_sms_clk	sel_sms_wdr	sel_sms_wir	sel_jpc_wdr	sel_jpc_wir	0	0	0	0	0	0	release_en
	W																
BISTCR5 ($BASE + 0x13)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	pg_cntl_vpu		pg_mode_vpu	pg_cntl_ipu		pg_mode_ipu	0	0	0	release_en
	W																
MBISTPASSR1 ($BASE + 0x16)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TVE_asap_fail_sms	GBIST_rom_failed
	W																
	R	SDMA_rom_bist_failed	GPU2D_reg_fail_sms	GPU2D_asap_fail_sms	VPU_reg_fail_sms	VPU_asap_fail_sms	L2data_fail_sms	L2cache_fail_sms	ETB_fail_sms	SIPMIX_star_fail_sms	SIPMIX_asap_fail_sms	LPMIX_reg_fail_sms	IPU_asap_fail_sms	0	GPU_star_fail_sms	GPU_reg_fail_sms	EMI_fail_sms
	W																

Table 8-1. SJC Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MBISTDONER1 ($BASE + 0x18)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TVE_asap_ready_sms	GBIST_rom_done
	W																
	R	SDMA_rom_bist_done	GPU2D_reg_ready_sms	GPU2D_asap_ready_sms	VPU_reg_ready_sms	VPU_asap_ready_sms	L2data_ready_sms	L2cache_ready_sms	ETB_ready_sms	SIPMIX_star_ready_sms	SIPMIX_asap_ready_sms	LPMIX_reg_ready_sms	IPU_asap_ready_sms	0	GPU_star_ready_sms	GPU_reg_ready_sms	EMI_ready_sms
	W																
MBISTMASKR1 ($BASE + 0x1A)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TVE_asap_bist_mask	GBIST_rom_mask
	W																
	R	SDMA_rom_bist_mask	GPU2D_reg_bist_mask	GPU2D_asap_bist_mask	VPU_reg_bist_mask	VPU_asap_bist_mask	L2data_bist_mask	L2cache_bist_mask	ETB_bist_mask	SIPMIX_star_bist_mask	SIPMIX_asap_bist_mask	LPMIX_reg_bist_mask	IPU_asap_bist_mask	0	GPU_star_bist_mask	GPU_reg_bist_mask	EMI_bist_mask
	W																

8.2.7 SJC Registers Description

The following registers are accessed using the extra debug register.

8.2.7.1 General Purpose Unsecured Status Registers 1,2,3 (Three Registers)

The General Purpose Unsecured Status Register 1 is a read-only register used to check the status of the different cores and the PLL. The rest of its bits are for general purpose use.

GPUSR1 — **General Purpose Unsecured Status Register 1** — **Addr $BASE + 0x00**

	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	0	0	0	0	0	0	PLL 3_lrf	PLL 2_lrf	PLL 1_lrf	0	0	0	0	0	A_WFI	A_DBG
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-2. General Purpose Unsecured Status Register 1

Table 8-2. General Purpose Unsecured Control Register Description

Name	Description	Settings
31:10 GPUSR1 Bits	reserved	
9 PLL3_lrf	status bit indicating if dpll3 is locked	0 = PLL not locked 1 = PLL lock
8 PLL2_lrf	status bit indicating if dpll2 is locked	0 = PLL not locked 1 = PLL lock
7 PLL1_lrf	status bit indicating if dpll1 is locked	0 = PLL not locked 1 = PLL lock
6:2 GPUSR1 Bits	reserved	
1:0 GPUSR1 Bits	SJC internal register - see SJC guide	

GPUSR2	General Purpose Unsecured Status Register 2													Addr $BASE + 0x01		
	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	0	0	0	0	0	0	0	0	0	0	0	0	sdma_debug_core_state[3:0]			
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-3. General Purpose Unsecured Control Register 2 Description

Table 8-3. General Purpose Unsecured Control Register 2 Description

Name	Description	Settings
31:4 GPUSR2 Bits	reserved	
3:0 sdma_debug_core_state[3:0]	SDMA core status (ipc_cstatus[3:0]). For whole bus value see GPUSR2 register.	0000 - PGM 0001 - DATA 0010 - Change Flow 0011 - Set Wakeup 0100 - DEBUG 0101 - FU Bus 0110 - SLEEP ... See SDMA spec for more details,

GPUSR3	General Purpose Unsecured Status Register 3													Addr $BASE + 0x02		
	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	0	0	0	0	0	0	0	ccm_ref_en_b	0	0	ccm_pdn_4all_req	ccm_pdn_4arm_req	system_in_stop_mode	system_in_wait_mode	ccm_ipg_stop	ccm_ipg_wait
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-4. General Purpose Unsecured Control Register 3 Description

Table 8-4. General Purpose Unsecured Control Register 3 Description

Name	Description	Settings
31:9 GPUSR3 Bits	Reserved	—
8 ccm_ref_en_b	indicates if external oscillator is enabled	0 = external oscillator disabled 1 = external oscillator is enabled
7:6 GPUSR3 Bits	reserved	—
5 ccm_pdn_4all_req	indicates a request to GPC to power down all units	0 = no power down request to GPC 1 = power down request sent to GPC
4 ccm_pdn_4arm_req	indicates a request to GPC to power down ARM	0 = no power down request to GPC 1 = power down request sent to GPC
3 system_in_stop_mode	indicates system is in stop mode	0 = system not in stop mode 1 = system in stop mode
2 system_in_wait_mode	indicates system is in wait mode	0 = system not in stop mode 1 = system in stop mode
1 ccm_ipg_stop	indicates CCM started stop entrance procedure	0 = CCM did not start stop entrance procedure 1 = CCM started stop entrance procedure
0 ccm_ipg_wait	indicates CCM started wait entrance procedure	0 = CCM did not start wait entrance procedure 1 = CCM started wait entrance procedure

8.2.7.1.1 Security Status Register

This register is used to reflect IC security status and is accessible in all the security modes. The assumption is that the information contained in this register does not pose any security breach for the system.

SSR — **Security Status Register** — **Addr $BASE + 0x06**

	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	iim_fuse_latched
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	0	~src_int_boot	ipt_secur_block	RSSTAT		SJM		FT	BSF	RSF	EBG	EBF	SWE	SWF	KTA	KTF
TYPE:	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-5. Security Status Register

Table 8-5. Security Status Register Description

Name	Description	Settings
31:17 SSR Bits	Reserved	—
16 iim_fuse_latched	Indicates that fuse values have been latched	0 = fuses have not yet been latched 1 = fuses have been latched
15 SSR Bit	Reserved	—
14 ~src_int_boot	SRC internal boot	—
13 ipt_secur_block	Indicates if invasive/non-invasive debug can be done to the ARM	0 = invasive and non-invasive debug cannot be done 1= invasive and non-invasive debug can be done
12:0 SSR Bits	SJC internal registers reserved—see SJC guide	—

8.2.7.2 General Purpose Clocks Control Register

This register is used to configure clock related modes in the i.MX51. These bits are directly connected to JTAG outputs.

GPCCR	General Purpose Clocks Control Register															Addr $BASE + 0x07
	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
TYPE:	jtag_sw_rst	dpdck0_2_en_pll3	pdf_pll3[3:1]			~pdf_pll3[0]	ref_clk_div_pll3	mfi_pll3[3:0]				dpdck0_2_en_pll2	pdf_pll2[3:1]			~pdf_pll2[0]
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Note:	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S	S
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	ref_clk_div_pll2	mfi_pll2[3:0]				dpdck0_2_en_pll1	pdf_pll1[3:1]			~pdf_pll1[0]	ref_clk_div_pll1	mfi_pll1[3:0]				SCLKR
TYPE:	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-6. General Purpose Clocks Control Register

Table 8-6. General Purpose Clocks Control Register Description

Name	Description	Settings
31 jtag_sw_rst	provides software reset to SRC	—
30 dpdck0_2_en_pll3	enables double frequency clock for dpllip3	0 = double frequency clock disabled 1= double frequency clock enabled
29:27 pdf_pll3[3:1]	pre division factor for dpllip3	0001 - PDF = 1 0000 - PDF = 2 0011 - PDF = 3 0010 - PDF = 4 0101 - PDF = 5 0100 - PDF = 6 0111 - PDF = 7 0110 - PDF = 8 1001 - PDF = 9 1000 - PDF = 10 1011 - PDF = 11 1010 - PDF = 12 1101 - PDF = 13 1100 - PDF = 14 1111 - PDF = 15 1110 - PDF = 16
26 ~pdf_pll3[0]	Note: This is the actual value of the pre-division factor. (after the PLL adds 1) See DPLL(IP) config spec for more details	
25 ref_clk_div_pll3	divides pll3 reference clock for dpllip3	0 = reference clock not divided by 2 1 = reference clock divided by 2

Table 8-6. General Purpose Clocks Control Register Description (continued)

Name	Description	Settings
24:21 mfi_pll3[3:0]	multiplication factor for dpllip3 If MFI is written a value less than 5, it will be defaulted to 5 This is also valid for ac scan mode	0000 = MFI is 5 0001 = MFI is 5 ... 0101 = MFI is 5 0110 = MFI is 6 0111 = MFI is 7 ...
20 dpdck0_2_en_pll2	enables double frequency clock for dpllip2	0 = double frequency clock disabled 1= double frequency clock enabled
19:17 pdf_pll2[3:1]	pre division factor for dpllip2	see above bits 29:26
16 ~pdf_pll2[0]		
15 ref_clk_div_pll2	divides pll2 reference clock for dpllip2	0 = reference clock not divided by 2 1 = reference clock divided by 2
14:11 mfi_pll2[3:0]	multiplication factor for dpllip2 If MFI is written a value less than 5, it will be defaulted to 5 This is also valid for ac scan mode	0000 = MFI is 5 0001 = MFI is 5 ... 0101 = MFI is 5 0110 = MFI is 6 0111 = MFI is 7 ...
10 dpdck0_2_en_pll1	enables double frequency clock for dpllip1	0 = double frequency clock disabled 1= double frequency clock enabled
9:7 pdf_pll1[3:1]	pre division factor for dpllip1	see above bits 29:26
6 ~pdf_pll1[0]		
5 ref_clk_div_pll1	divides pll1 reference clock for dpllip1	0 = reference clock not divided by 2 1 = reference clock divided by 2
4:1 mfi_pll1[3:0]	multiplication factor for dpllip1 If MFI is written a value less than 5, it will be defaulted to 5 This is also valid for ac scan mode	0000 = MFI is 5 0001 = MFI is 5 ... 0101 = MFI is 5 0110 = MFI is 6 0111 = MFI is 7 ...
0 SCLKR	SJC internal register - see SJC guide	

8.2.7.2.1 General Purpose Unsecured Control Registers 1,2,3 (Three Registers)

These registers are used to configure WMSG IEEE 1149.1 JTAG for special test or debug modes (see the test guide for more information). These registers are not secured (accessible in all JTAG security modes). The bits of these registers are directly connected to SJC outputs.

GPUCR1	General Purpose Unsecured Control Register														Addr $BASE + 0x09	
	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	efuse_prog_supply_gate	0	0	0	0	ipt_sjc_test_mode[1:0]		ipt_sjc_lch_mode[2:0]			0	0	0	0	0	0
TYPE:	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	0	0	0	0	0	0	0	0	0	ipt_bipg_tdien	0	0	0	0	usbphy_source_sel	0
TYPE:	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-7. General Purpose Unsecured Control Register

Table 8-7. General Purpose Unsecured Control Register 1 Description

Name	Description	Settings
31 efuse_prog_supply_gate	Supply gating for eFUSE programing	0 = fuse programing supply voltage is gated off to the efuse module 1 = allow fuse programing
30:27 GPUCR1 Bits	reserved	
26:25 ipt_sjc_test_mode[1:0]	These bits determine the type of burn_in mode	00 = BI_BIST 01 = BI_LCH_ALL
24:22 ipt_sjc_lch_mode[2:0]	Long Chain configuration	000 = ALL_LONCHAIN 001 = ARM_LONG_CHAIN 010 = APU_LONG_CHAIN 100 = MIXES_LONG_CHAIN 101 = BIPG_ONLY_LONG_CHAIN 110 = NO_LONG_CHAIN
21:11 GPUCR1 Bits	reserved	
6 ipt_bipg_tdien	This bit enables tdi for BURN IN mode	0 = tdi not enabled in BURN IN mode 1 = tdi enabled in BURN IN mode
5:2 GPUCR1 Bits	reserved	

Table 8-7. General Purpose Unsecured Control Register 1 Description

Name	Description	Settings
1 usbphy_source_sel	Selects USBPHY input source	0 = USBOH3 1 = external source through gpio
0 GPUCR1 Bits	reserved	

GPUCR2 **General Purpose Unsecured Control Register 2** **Addr $BASE + 0x0A**

	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	0	0	init_ssi_apm_clk_sel		init_periph_clk_sel	~init_periph_apm_sel[1]	init_periph_apm_sel[0]	~init_nfc_podf[2]	init_nfc_podf[1:0]		~init_emi_slow_podf[2]	init_emi_slow_podf[1:0]		~init_axi_b_podf[2]	init_axi_b_podf[1]	~init_axi_b_podf[0]
TYPE:	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	init_axi_a_podf[2]	~init_axi_a_podf[1:0]		~init_ahb_podf[2]	init_ahb_podf[1:0]		init_ipg_podf[1]	~init_ipg_podf[0]	~init_perclk_podf[2]	init_perclk_podf[1]	~init_perclk_podf[0]	init_perclk_pred2[2:0]			init_perclk_pred1[1]	~init_perclk_pred1[0]
TYPE:	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-8. General Purpose Unsecured Control Register 2

Table 8-8. General Purpose Unsecured Control Register 2 Description

Name	Description	Settings
31:30 GPUCR2 Bits	reserved	
29:0	These bits connected to CCM (clock manager). They define initialization values to various clock MUX selects and clock dividers, thus defining initial system clocks frequencies. See CCM guide for more details.	

GPUCR3	General Purpose Unsecured Control Register													Addr $BASE + 0x0B	
BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	sjc_ipt_io_tail_bypass	sjc_interrupt
TYPE: rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
arm_clk_off_on_lpm	eim_d15_alt7_hard_en	en_tester_cntl	smart_bypass	0	sjc_sre	sjc_sr_fast	sjc_sr_slow	sjc_dse_en	sjc_dse1	sjc_dse0	ahb_clk_bypass	emi_slow_clk_bypass	axi_a_clk_bypass	0	0
TYPE: rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
RESET: 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8-9. General Purpose Unsecured Control Register 3

Table 8-9. General Purpose Unsecured Control Register 3 Description

Name	Description	Settings
31:18 GPUCR3 Bits	Reserved	—
17 sjc_ipt_io_tail_bypass	Bypasses the FF at the output to the BGA contact	0 = Flip flop at BGA contact output not bypassed 1 = Flip flop at BGA contact output bypassed
16 sjc_interrupt	This interrupt allows the system to exit stop mode while the chip is in debug mode	—
15 arm_clk_off_on_lpm	Gates arm clock	0 = ARM clock not gated 1 = ARM clock gated
14 eim_d15_alt7_hard_en	alt7_hard_en for IOMUX cell EIM_D15	0 = EIM_D15 BGA contact is not hard_en in alt7 1 = EIM_D15 BGA contact is hard_en in alt7
13 en_tester_cntl	Allows tester to receive determinism of reset seq	0 = Tester cannot receive determinism of reset sequence 1 = Tester receives determinism of reset sequence
12 smart_bypass	Memory repair bypass during POR - functional mode	0 = Memory Repair is not bypassed 1 = Memory Repair is bypassed
11 GPUCR3 Bit	Reserved	—
10 sjc_sre	sjc slew rate controller for IOMUXC	0 = Slow slew rate 1 = Fast slew rate
9 sjc_sr_fast	sjc slew rate fast for IORING - this bit must be in coordination with sjc_sre	0 = Fast slew rate not enabled 1 = Fast slew rate enabled

Table 8-9. General Purpose Unsecured Control Register 3 Description (continued)

Name	Description	Settings
8 sjc_sr_slow	sjc slew rate slow for IORING - this bit must be in coordination with sjc_sre	0 = Slow slew rate not enabled 1 = Slow slew rate enabled
7 sjc_dse_en	sjc BGA contact strength enable	0 = Drive Strength value does not come from SJC 1= Drive Strength value comes from SJC
6 sjc_dse1	sjc BGA contact strength	00 = Low Drive Strength 01 = Medium Drive Strength 10 = High Drive Strength 11 = Max Drive Strength
5 sjc_dse0	sjc BGA contact strength	
4 ahb_clk_bypass	Use bypass clock for AHB root in CCM	0 = regular 1 = bypass from BGA contact
3 emi_slow_clk_bypass	Use bypass clock for EMI_SLOW root in CCM	0 = regular 1 = bypass from BGA contact
2 axi_a_clk_bypass	Use bypass clock for AXI_A root in CCM	0 = regular 1 = bypass from BGA contact
1:0 GPUCR3 Bits	Reserved	—

8.2.7.3 Product ID and JTAG ID

i.MX51 PROD_ID = 11110010 and JTAG_ID = 0x0190_C01D. Product revision (hw_rev[7:4] in IIM) is = 4’b0000 accordingly.

8.3 CoreSight Design Kit

The i.MX51 includes an ARM CoreSight component for multicore debug and trace solution. CoreSight component can be found in the following hierarchy levels: ARM core, ARM platform, and top level.

- ARM core—include the following CoreSight components: ETM, CTI0
- ARM platform—include the following CoreSight components: ETB, ATB Replicator, CTI1, CTM
- Top level—include the following CoreSight components: DAP, CTI2, CTI3, TPIU

8.3.1 Memory Map and Register Definition

Each CSDK component has a 4 Kbyte location block in the CoreSight memory map. The base address of the CoreSight location block is not fixed but the address offsets are fixed. Each CSDK component has a 4-Kbyte memory map, that is, 1 Kbyte words.

Components connected to the DAP internal bus appear as part of a memory mapped structure with parallel address and two data buses, read data and write data. The bus master, JTAG-DP, uses normal bus transactions to control the various APs, in this structure the slaves. The JTAG-DP reads and writes registers within this bus.

8.3.2 CoreSight Clock Enable

By default the CoreSight component clocks are enabled when the system exists from reset with all clocks enabled. The CoreSight clocks' gating is controlled by programmable CCM registers and DBGEN. The CCM allows control of each CoreSight component separately (by default) and the DBGEN controls all debug clocks. DBGEN is raised when detecting activity on JTAG ports (TMS) and descend on reset (by dap_sys) or controlled by ARM. Only if both CCM registers bits and DBGEN are high the clock will be propagate to CoreSight component.

8.3.3 CoreSight Debug Access Port (DAP) and DAP_SYS

ARM's Debug Access Port (DAP) has several functions. DAP_SYS is a wrapper around DAP and is the FSL block that is actually instantiated in a design (DAP is contained within DAP_SYS). The following list summarizes the features provided by DAP_SYS.

- Enables debug-related communication between various parts of the system
  - It is a modular block.
  - It has slave-side support for JTAG and APB (ARM Peripheral Bus) protocols.
  - The Debug master can access many debug resources in real time without having to halt the core.
  - It can enable system access to anything connected to the Debug-APB via the APB multiplexor.
- ROM Table provides a list of memory locations of CoreSight components connected to the Debug APB. Visible from both tools and system access.
- AHB to APB gasket to translate System AHB accesses to APB (input as system APB to the DAP).
- APB Decode to issue select signals to CoreSight components on APB, and handle the PSLVERR and PREADY signals from the components to the DAP.
- AHB-Lite to AHB converter, to translate DAP's AHB-lite accesses to system's AHB protocol.

Debug and Reset logic generate DBGEN signal and reset signals. The CoreSight components that reside on the Debug-APB include all Embedded Cross Trigger blocks, the ETM, and the ETB.

8.3.4 Embedded Cross Trigger (ECT)

System events, such as a debug request, a core entering or exiting debug mode, the occurrence of a breakpoint or watchpoint, the occurrence of a particular error, the state of a buffer, etc., can be useful or even essential for debugging or profiling system performance. Whatever the source of the event, the embedded cross trigger implements a flexible, programmable mechanism to transport these events from a source to one or multiple destinations. This includes handling handshake requirements with both the source and the destination as needed and synchronization of signals from different clock domains. The end result is that events of interest that occur in one domain, such as in the Cortex-A8 domain, can be recognized and responded to be a destination domain, such as the SDMA.

The i.MX51 uses the CoreSight ECT (provided by ARM as part of their full CoreSight Debug Support package) and a custom wrapper to accommodate the various necessary interface scenarios. The ECT (as delivered by ARM) consists of two major blocks—a central Cross Trigger Matrix (CTM) and a Cross Trigger Interface (CTI) block. A Freescale-custom wrapper is added to the CTI to accommodate a variety

of different input and output interface scenarios. The wrapped CTI is called the Extended CTI and is configured on a signal by signal basis with port tie-off's.

i.MX51's ECT architecture consists of a single CTM and three wrapped CTI's covering three basic groups of functionality as follows:

- CTI0 is in the ARM Cortex A8 core.
- CTI1 is in the Cortex-A8 core platform, with two triggers in and two triggers out used by the platform and the others by the device peripherals.
- CTI2 is in the SOC level and is SDMA.
- CTI3 is in the SOC level and is used by MCU peripherals (IPU, GPU, VPU).

A detailed CTI pin assignment can be found in on page 8-21.

Figure 8-10 is a simple illustration of the relationship between the two processor domains, the SJC and peripherals, and the ARM CoreSight DAP. APB is an AMBA bus used for debugging. It defines what debug and trace components are required and how they are connected. The DAP is the Debug-APB master for the following debug units: ETM, ETB, TPIU, and the ARM Cortex A8 debug unit, which are four CTI blocks. The Debug-APB bus differs from system-level APBs in that it can be directly accessed via JTAG as well as from the ARM core. Module selects for each slave module on the Debug-APB are generated by the DAP_SYS.

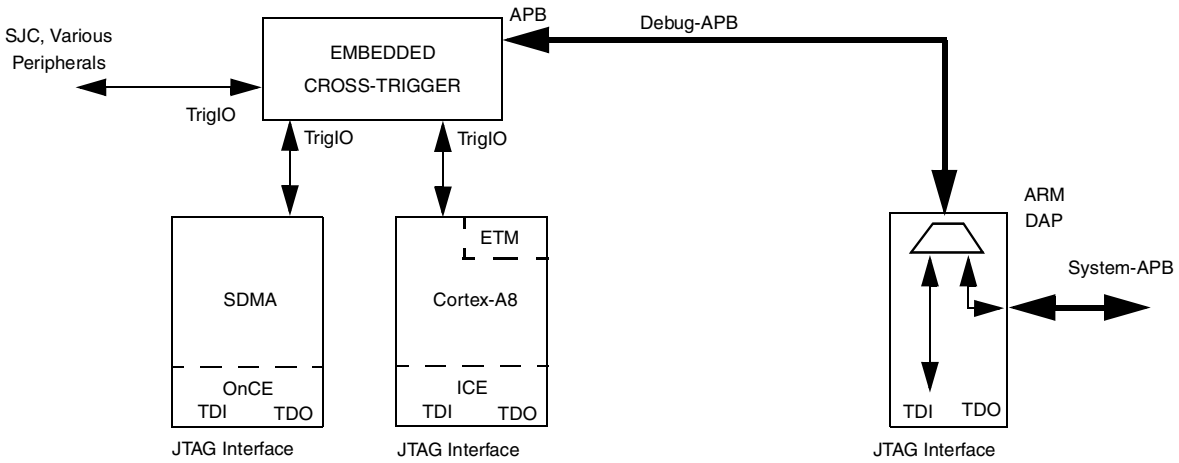


Figure 8-10. Embedded Cross Trigger Topology

Figure 8-11 illustrates the connectivity between the CTM and CTIO components of the CoreSight ECT.

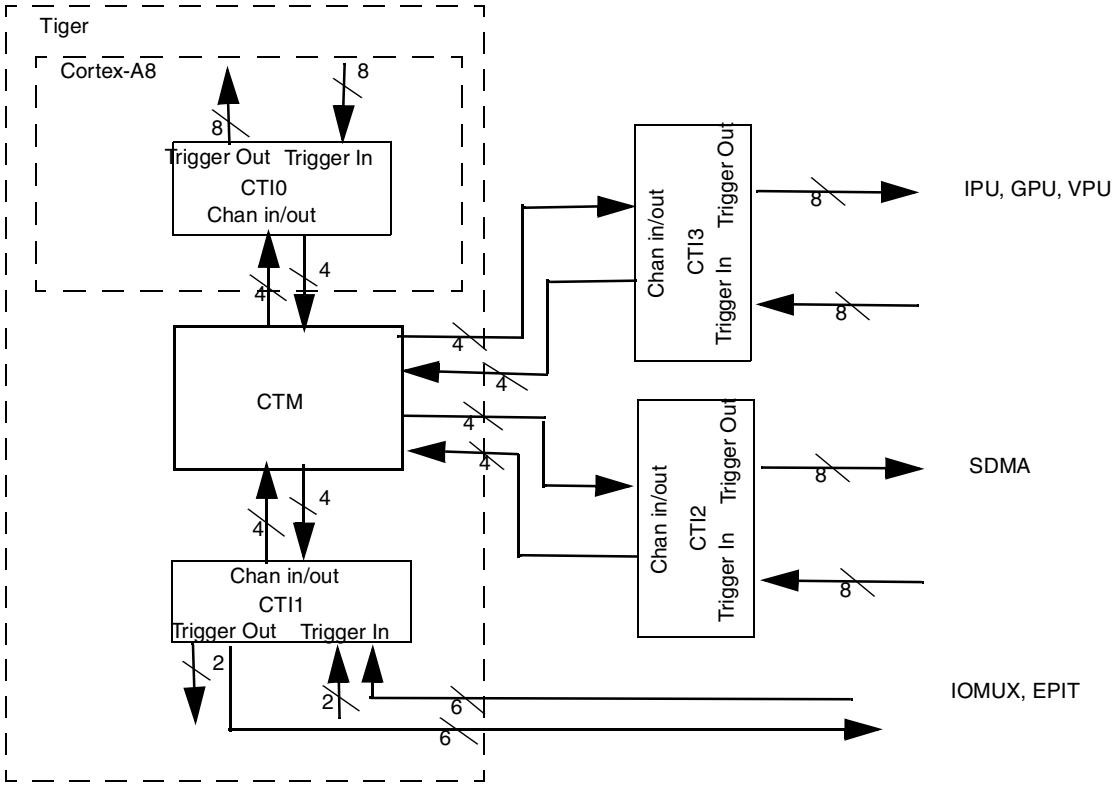


Figure 8-11. Embedded Cross Trigger - Basic Architecture

8.3.4.1 CoreSight CTM

The CTM (Cross Trigger Matrix) is provided by ARM. The CTM is a relatively simple block with no programmability. On i.MX51, it is configured to have four input and four output channels. Whatever comes in on an input channel is routed to all corresponding output channels. For instance, if a trigger input comes in on input channel 0 from CTI0, it is routed to output channel 0 to CTI1, CTI2, and CTI3. All selection of drivers and receivers for an event is done in the individual CTI blocks.

The final feature provided by the CTM is handshake with each CTI. This insures that as trigger signals cross different clock domain boundaries, they will not be lost due to slower downstream sampling frequencies. It is a simple mechanism in which a CTI holds an output until it receives the acknowledgement from the CTM. Handshake is also implemented in the reverse direction, insuring that a trigger signal propagating from the CTM to a CTI is not missed by the CTI.

8.3.4.2 CoreSight CTI

The CTI (Cross Trigger Interface) is also provided by ARM. A summary description of the CTI is provided here, but please refer to ARM documentation for detailed information. As mentioned earlier, there are three CTIs. Each of these has eight trigger inputs and eight trigger outputs that connect to logic in the domain to be debugged or profiled. Each CTI also includes a 4-channel interface to the CTM (four inputs

and four outputs). The CTIs are wrapped with additional logic to from the Extended CTI, which is described in the next section.

“

Table 8-10. CTI 1 Input Assignments

Trigger input	Debug Resource Name	Clock[1]	Channel Function	SYNC (dbg_atclk)	Active	ack	Comment
0	TBD	lpcg_ahb_clk_tzic	Configured interrupt	No	High	No	
1	TBD	lpcg_ahb_clk_tzic	Configured interrupt	No	High	No	
2–3		—	TRACING	—	—	—	ARM
4	TBD	lpcg_ahb_clk_tzic	Configured interrupt	No	High	No	
5	TBD	lpcg_ahb_clk_tzic	Configured interrupt	No	High	No	
6	TBD	lpcg_ahb_clk_tzic	Configured interrupt	No	High	No	
7	cti_in[7]	—	Debug	—	—	Yes	IOMUX

1 The “Clocks” column specifies the associated clock that needs to be running for the event to propagate.

“

Table 8-11. CTI 1 Output Assignments

Trigger Output	Debug Resource Name	Clock[1]	Channel Function	SYNC (dbg_atclk)	Active	ack	Comment
0	intin_b	lpcg_ahb_clk_tzic	General Purpose	No	—	No	Allows creating interrupts
1	intin_b	lpcg_ahb_clk_tzic	General Purpose	No	—	No	Allows creating interrupts
2	intin_b	lpcg_ahb_clk_tzic	General Purpose	No	—	No	Allows creating interrupts
3	intin_b	lpcg_ahb_clk_tzic	General Purpose	No	—	No	Allows creating interrupts
4–5	—	arm_clk	—	—	—	—	ARM
6	cti_out[6]	—	Debug	No	—	—	IOMUX
7	cti_out[7]	—	Debug	No	—	—	IOMUX

1 The “Clocks” column specifies the associated clock which need to be running for the event to propagate.

Table 8-12. CTI 2 Input Assignments

Trigger Input	Debug Resource Name	Clock[1]	Channel Function	SYNC (debug_apb_clk)	Active	ack	Comment
0	debug_mode	ipg_clk_spba	Debug Acknowledge	Yes	High	No	SDMA
1	debug_core_run	ipg_clk_spba	debug	Yes	High	No	SDMA
2	evt_chn_lines[0]	ipg_clk_spba	debug	Yes	High	No	SDMA
3	evt_chn_lines[1]	ipg_clk_spba	debug	Yes	High	No	SDMA
4	evt_chn_lines[2]	ipg_clk_spba	debug	Yes	High	No	SDMA
5	req_ma	ipg_clk_spba	debug	Yes	High	No	SPBA
6	req_mb	ipg_clk_spba	debug	Yes	High	No	SPBA
7	req_mc	ipg_clk_spba	debug	Yes	High	No	SPBA

1 The “Clocks” column specifies the associated clock which need to be running for the event to propagate.

Table 8-13. CTI 2 Output Assignments

Trigger Input	Debug Resource Name	Clock[1]	Channel Function	SYNC (debug_apb_clk)	Active	ack	Comment
0	events[31]	ipg_clk_sdma		Yes	High	No	SDMA
1	sdma_dreq_i	ipg_clk_sdma		Yes	High	No	SDMA
2	FLUSHIN	lpcg_debug_clk_tpiu		No	High	Yes	TPIU, FLUSHINACK
3	TRIGIN	lpcg_debug_clk_tpiu		No	High	Yes	TPIU, TRIGINACK
4	system_debug	GND/tck		Yes	High	No	SJC
5	emi_spare_ports_in[0]	GND		Yes	High	No	EMI
6							Not used
7							Not used

1 The “Clocks” column specifies the associated clock which need to be running for the event to propagate.

Table 8-14. CTI 3 Input Assignments

Trigger Input	Debug Resource Name	Clock[1]	Channel Function	SYNC (debug_apb_clk)	Active	ack	Comment
0	ipi_int_epit_oc	ipg_clk_epit1	TRACING	Yes	High	No	EPIT
1	ipi_int_ipu_func	hsp_clk	debug	Yes	High	No	IPU general interrupt
2	ipi_int_ipu_err	hsp_clk	debug	Yes	High	No	IPU error interrupt
3	—	—	—	—	—	—	—

Table 8-14. CTI 3 Input Assignments (continued)

Trigger Input	Debug Resource Name	Clock[1]	Channel Function	SYNC (debug_apb_clk)	Active	ack	Comment
4	gpu_use_bufid	ack_gpu	debug	Yes	High	No	GPU - OR output of the bus 3 bits
5	gpu_int_b	ack_gpu	debug	Yes	Low	No	gpu genral perpose interrupt
6	vpu_idle	cclk	Trace	Yes	High	No	VPU Idle
7	vpu_underrun	cclk	debug	Yes	High	No	VPU Underrun

1 The “Clocks” column specifies the associated clock which need to be running for the event to propagate.

Table 8-15. CTI 3 Output Assignments

Trigger input	Debug Resource Name	Clock	Channel Function	SYNC (debug_apb_clk)	Active	ack	Comment
0	—	—	—	—	—	—	Not used
1	—	—	—	—	—	—	Not used
2	—	—	—	—	—	—	Not used
3	—	—	—	—	—	—	Not used
4	—	—	—	—	—	—	Not used
6	—	—	—	—	—	—	Not used
7	—	—	—	—	—	—	Not used

8.3.4.3 Extended CTI (CTI Wrapper)

This wrapper consists of additional logic for each input and each output trigger to implement several features that can be selected or enabled with hard tie-off’s on the module’s boundary. It is combined with ARM’s CTI block to form the CTI_Extended block. A summary of the extended input features implemented by logic in the wrapper is as follows:

- Invert the incoming signal
  - The ARM CTI assumes all inputs are active-high. An active-low input must first be inverted before going into the ARM CTI
- Sample the incoming trigger signal with a clock synchronous to the incoming signal
  - Samples the incoming trigger with its own clock
- Convert the incoming signal to a pulse
  - Used when an input trigger has a long assertion time. Destinations may be sensitive to a long assertion time and may see it as multiple input triggers.
- Implement holding logic to insure that the ARM CTI captures and acknowledges the incoming signal. The holding logic clock must be synchronous to the clock that generated the input trigger.

- Provides a synchronized or asynchronous acknowledge signal back to the origin of the trigger.
  - If the asynchronous trigger source requires an ACK signal back from the CTI, but a clock from that source is not available to the CTI, the source must first synchronize the received ACK before using it in the trigger source logic.

8.3.5 CoreSight Trace Port Interface (TPIU)

The TPIU is one of the CoreSight trace sink components it acts as a bridge between the on-chip trace data to a data stream that is then driven out the trace port. ATB interface is used by the TPIU accepts trace data from a trace source, either direct from a trace source or using a Trace Funnel. The TPIU has 32 bit connected to the chip BGA contact. The APB interface is the programming interface for the TPIU configuration.

8.4 ARM Cortex A8 Core and Platform

ARM Cortex A8 Debug architecture includes support for TrustZone and CoreSight. The memory-mapped external debug interface replaces the coprocessor interface defined in the previous version of the Debug architecture. A full access to the processor debug capability available by ARM Cortex A8 debug register map through the *Advanced Peripheral Bus* (APB) slave port. The core includes Processor Debug Unit allow: stop program execution, examine and alter processor and coprocessor state, examine and alter memory and input/output peripheral state, and restart the processor core.

8.4.1 ARM Cortex A8 Core Debug Support Features

The ARM Cortex A8 core debug support features are as follows:

- CoreSight Embedded Trace Macro (ETM11)—trace generator for the ARM Cortex A8 core
- Support for a TrustZone-relateyd 3-level debug scheme:
  - Debug everywhere
  - Debug in Non-Secure privileged and user and Secure user
  - Debug in Non-Secure only
- EmbeddedICE-RT logic
  - Support for both monitor-mode and halt-mode debugging.
  - Core run/halt control, debug status/control
  - Breakpoint/watchpoint control
  - Core- and memory-mapped resource examination/modification
- Data communication channel between ARM core and host debugger via JTAG
- PMU—Performance Metrics Unit used for system profiling and debug.
- CP15 register for debugging the MMU, I and D L1 cache, and TLB
- EVTMON-L2 Event Monitor—Supports debug and profiling of the ARM’s L2 cache activity

8.4.2 Additional Platform Debug Functionality

- CoreSight Embedded Trace Buffer (ETB11)—4-Kbyte RAM array to be used for on-chip capture of trace data output from the ETM11
- ATB Replicator to connect the trace data to TPIU (Trace Port Interface) and ETB (Embedded Trace Buffer).
- Debug Visibility—select critical signals routed to the I/O pads as alternate outputs for external visibility

8.5 Smart DMA (SDMA) Core

The SDMA is a dedicated, programmable DMA engine. It is an integration of a 32-bit RISC core and DMA-specific hardware, and it includes ports for the AP domain and a peripheral domain, along with a burst-capable port for direct external memory access. The SDMA and its integration in the i.MX51 is unchanged from previous SOCs.

The main SDMA debug features are as follows:

- OnCEOn Chip Emulator, provides the following capabilities:
  - SDMA core control - run/halt/single-step
  - SDMA core register/memory-map access
  - Event detection, watchpoints, and hardware breakpoints
  - Real time buffer and PC trace buffer capability
- Trace buffer
  - Contains information to identify the 32 last changes of flow detected during a program execution
- Context dump
  - include information about all the channel dump activity
  - Current contents of SDMA RAM
- ROMPATCH

8.5.1 SDMA On Chip Emulation Module (OnCE) Feature Summary

The SDMA debug features are primarily defined by the OnCE portion of its design, which are summarized as follows:

- Memory And Register Access—dedicated logic enables user-access to SDMA memory and register locations. These accesses are supported only when the processor is in debug mode.
- Event Detection Unit—watches signals from the data memory bus (DMBus) which is used by the RISC core to access its RAM, ROM, and memory-mapped registers
- Watchpoints—one output signal is available to watch event matching conditions at the chip level. Match conditions are defined by programming memory-mapped registers.
- Hardware Breakpoint—a counter is decremented after an event detection. A debug request is sent to the SDMA core only when the counter reaches the value of zero. It is possible to program the

initial value of the counter or to disable the use of the counter if a debug request must be generated after each event detection.

- Real Time Buffer—The Real Time Buffer Register (RTB) is a single 32-bit memory-mapped register which can be accessed as a regular memory location during program execution. It is used to store and retrieve run time information without putting the SDMA in debug mode. Each write to this register causes an event. This register is, in fact, located in the OnCE. Executing through JTAG, a buffer command exports the content of this register through the JTAG port.
- Core Control (Core Status/Single Stepping)—Commands are provided to monitor and control processor activity. The commands can halt the core, rerun the core from another address location, and get processor status.
- Trace Buffer—a 32 × 32 buffer that records the last 32 changes of flow during program execution. The buffer stores data in a modulo fashion (i.e. the 33rd instruction change replaces the 1st). Captured trace information is retrieved via reads to the Trace Buffer Register.

8.5.2 Other SDMA Debug Functionality

Other SDMA debug functionalisty is as follows:

- Core Trace—basic core trace capability is available through debug visibility functionality only. ETM/Nexus trace capability does not exist.
- ROM Patch—can be accomplished by manipulating the CHN0ADDR register through JTAG or via the MCU's ability to write to SDMA OnCE registers. This must be done right after reset and before the SDMA core is enabled to begin processing events.
- Additional debug control/status interaction with the SJC module
  - SJC-controlled Debug Request
  - SJC-readable Debug Acknowledge (in debug mode)
  - Debug clock control - allows SJC to force clocks on for debug purposes
  - Debug core state (SDMA RISC Core State) - 4 bits accessible from the SJC via JTAG
- Debug Visibility—observable outputs as alternate (programmable) output functions of I/O pins
  - Debug Request, Debug Mode
  - Debug Yield
  - Debug Event Channel[5:0] (indicates requesting event or channel being processed)
  - Debug PC [13:0] (for SDMA core trace)
  - Debug core state [3:0]
  - Debug Real-time Buffer write
  - Debug Addr/Data match
  - Debug bus error
  - Debug bus device
  - Debug bus r/w
  - Debug Event Channels[7:0]
  - Debug Core Run (active when core is running)

8.5.3 Embedded Cross Trigger Interface

Please refer to the Embedded Cross Trigger section of this chapter for detailed information about the SDMA interface to that subsystem.

8.6 External Memory Interface (EMI)

The second-generation EMI contains an arbitration profiling unit. This can be used to monitor and profile the dynamic arbitration behavior during operation. Several internal signals would be routed out by EMI debug unit to give SoC the capability to track the internal logic mainly for the arbitration mechanism and memory served accesses. In addition, the debug unit gives the ability to profile a master connected to EMI and get the bus performance of the arbitration.

Signals routed to 51 bit **ipp_do_emi_debug** bus are routed to the BGA contact.

The Profiling unit includes the following:

- A register that saves the maximal time that a selected master was pending without getting the bus (MDSR0)
- The maximum value of the dynamic priority which was selected (MDSR0)
- Counters to record the number of access that each master performed through each channel (MDSR2–5)
- Counter of the total number of accesses a specific request (or type of request) has accessed the bus can be fast, slow or intr (MDSR6)
- A register that sums of "time for bus" the time it took a specified request (or type of request) to get the bus (MDSR7)
- A register that sums the time it took each access to get the bus (pending time) (MDSR8)
- Ability to reset all the counters

The EMI profiling logic includes a START signal, which comes from the ECT. Thus, any definable ECT condition can be used to initiate profiling.

8.7 Debug Visibility—IOMUX

Certain predefined, internal signals can be viewed at the package pins. The BGA contact logic for most pins includes an IOMUX, allowing that BGA contact to be shared across multiple functions. Each BGA contact has a primary function that is selected at reset. The alternate functions, such as displaying the state of an internal signal for debug purposes, is selected by reprogramming the IOMUX.

Table 8-16 shows the debug visibility.

Table 8-16. Debug Visibility

Module	Signals
SDMA	debug_mode, debug_bus_error, debug_bus_device[4:0], debug_bus_rwb, debug_matched_dmbus, debug_rtbuffer_write, debug_evt_chn_lines[7:0]
EMI	ipp_do_emi_debug[50:0]
IPU	ipu_diagbus[15:0]
GPU	SYS_GC_debug_out[16:0]
TPIU	TRACEDATA[31:0]

8.8 MCU Peripherals

This section discusses the following MCU peripherals:

- Section 8.8.1, Image Processing Unit (IPU)”
- Section 8.8.2, Video Processing Unit (VPU)”
- Section 8.8.3, Graphics Processing Unit (GPU)”

8.8.1 Image Processing Unit (IPU)

The IPU include debug unit that allow routing of its signals to the SoC level. The signals muxing to be routed to top are controlled by DP Debug Control register (DP_DEBUG_CNT) and are connected to 16 bit diagnostic bus (ipu_diagbus). The IPU also include status register (DP_DEBUG_STAT) that can be used for debug.

The IPU can enter the system into debug mode by the cross trigger system. Three of the IPU signals are input to ETC: IPU general interrupt, IPU error interrupt, and IPU end of frame. These inputs can trigger the system to enter debug mode.

More details on signal muxing and IPU debug registers can be found in IPU chapter.

8.8.2 Video Processing Unit (VPU)

The VPU can enter the system into debug mode by the cross trigger system. Three of the VPU signals are input to ETC: VPU idle, VPU interrupt, and VPU underrun. Those signals can trigger the system to enter debug mode.

VPU internal register can access the AHB bus for further analysis.

8.8.3 Graphics Processing Unit (GPU)

The GPU supports propagating debug information from the core over a 32-bit wide debug bus. The debug outputs are directed to only 16 visible ports. GPU includes debug control bus SYS_GC_gpio[15:0] that selects which debug signals are exposed on the debug out bus GC_SYS_debug_out[31:0].

GPU can enter the system into debug mode by the cross trigger system. Two of the GPU signals are input to ETC: GPU error interrupt and GPU end of frame. Those signals can trigger the system to enter debug mode.

8.9 Supported Tools

The i.MX51 supports RealView™ ARM Debugger, the debugger should be connect to i.MX51 from host by RealView ICE protocol converter.

8.10 Interrupt Visibility

Similar to debug visibility, the i.MX51 includes multiplexors that allow users to view selected internal interrupts. Up to five pads support this as alternate BGA contact output functions, allowing the user to view up to five different interrupt sources simultaneously. These five signals also route to ECT inputs, allowing them to generate ECT trigger events as desired.

8.11 Miscellaneous

This section discusses the SOC-level bus trace and clock/reset/power block interactions.

8.11.1 SOC-level Bus Trace

There is no SOC-level bus trace capability on i.MX51

8.11.2 Clock/Reset/Power

The interactions between the debug system and the system clock, reset, and power control blocks include enhancements to insure that status of critical system components can be monitored at all times and that a debugger can insure that all critical clocks and power are enabled when needed. Specifically, a master signal coming from an SJC control bit and going to the Clock/PLL logic is implemented for each major clock domain. The same is done for each power domain. Historically, JTAG based operations have not been possible under certain system conditions. For the i.MX51, the debugger must have the ability to determine the status of all critical system elements and to override any condition that would prevent the desired debug resources from operating properly.

Chapter 9
System Boot

9.1 Introduction

The i.MX51 allows the user to configure a wide variety of boot configurations for both engineering development and production. The boot ROM logic supports features such as booting from various external memory devices, support for downloading code through a serial downloader, boot device configuration, and secure boot. This chapter describes how the boot process works and how to program it.

The boot process begins at Power On Reset (POR). The boot ROM logic reads hardware inputs such as the boot pins on the IC, eFUSEs, and/or GPIO settings to determine the boot flow behavior of the i.MX51.

The default out-of-reset boot sequence is a High Assurance Boot (HAB) secure boot environment. The HAB is a combination of hardware and software combined with a PKI (Public Key Infrastructure) protocol to protect the system from executing unauthorized images or programs. Before the HAB allows the user's boot code to execute, the code must be signed by the private key holder, which is then compared with the public key on i.MX51. The HAB library in the i.MX51 boot ROM also provides a number of API functions allowing the user to authenticate any defined region and signature at run-time.

For non-secure operations and testing, the i.MX51 also allows the HAB to be bypassed while reading the internal ROM or using a direct external boot allowing the processor to boot directly from external memory, as done by traditional microprocessors.

The boot ROM logic also allows the downloading of and the ability to flash new ROM code via a serial connection. Typically, the serial downloader application is downloaded to internal RAM, which facilitates the ROM Flash programming. The download uses either a high-speed USB in nonstream mode or a UART connection.

The boot capabilities can differ substantially depending on the HAB type security configuration being employed and the other boot configuration settings.

The i.MX51 supports two general configuration environments. Full boot flexibility is supported in the development (or engineering) configuration but in the production (or secure) configuration significant limitations on the boot process exist. The remainder of this chapter provides the details about how to configure and use the boot features of the i.MX51.

9.2 Boot Module Activation

The i.MX51 boot logic affects up to 17 different hardware modules, which are activated and play a vital role in the boot flow. The processor configures and uses the following modules (listed in alphabetical order) during the boot process:

- CCM—Clock Control Module

- CSU—Central Security Unit. The Security Control Registers (SCR) of the CSU are set during boot time by the High Assurance Boot (HAB) code and are locked to prevent further writing.
- eCSPI—enhanced Configurable Serial Peripheral Interface
- EMI[1] (WEIM/NFC/ESDCTL)—External Memory Interface
- eSDHC—enhanced Secure Digital Host Controller
- HS-I^{2}C—High Speed Inter IC
- I^{2}C—Inter IC
- IIM—IC Identification Module. The IIM contains the eFUSEs.
- IOMUX—I/O Multiplexor allows using the GPIO to override eFUSE boot settings.
- PLL—Phase Locked Loop. Also called Digital Phase Locked Loop (DPLL)
- RTIC—Run Time Integrity Checker. The RTIC's purpose is to ensure the integrity of the peripheral memory contents, protect against unauthorized external memory elements replacement, and assist with boot authentication.
- SAHARA—Symmetric/Asymmetric Hash And Random Accelerator
- SCC—Security Controller
- SRC—System Reset Controller. The SRTC incorporates a special System State Retention Register (SSRR) that stores system parameters during system shutdown modes including the value of the BMOD pins
- SRTC—Secure Real Time Clock
- UART and USB—used for serial download of new ROM code to Flash the Boot ROM
- WDOG—Watchdog Timer

1. Only used for booting from an external memory device using the WEIM, NFC, or eSDCTL.

9.3 Internal ROM /RAM Memory Map

Figure 9-1 shows the internal ROM/RAM memory map.

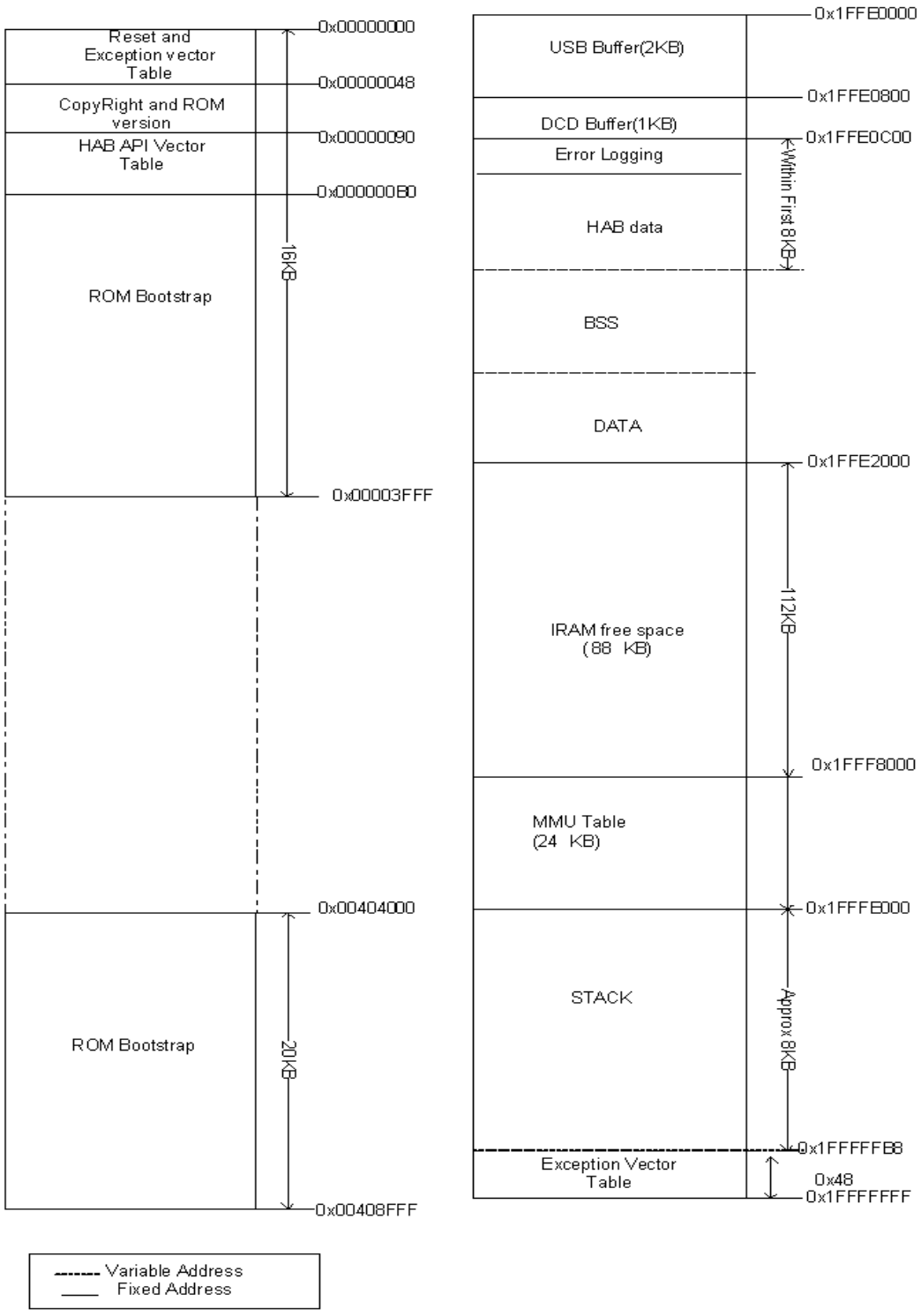


Figure 9-1. Internal ROM and RAM Memory Map

9.4 Boot Modes

The i.MX51 has four boot modes (one is reserved for internal use) that are selected by the two boot mode contacts on the IC package (BOOT_MODE0/1]). The settings of these two contacts are sampled upon exit of reset and stored in the BMOD[1:0] field of the SRC Boot Mode Register (SBMR) register in the SRC (System Reset Controller) module. Connecting the Boot Mode contacts to GND reads as a logic 0. For logic 1, Freescale recommends tying the contact to NVCC_PER3

The boot modes are: internal, internal boot with fuses, and serial boot via USB/UART. Refer to Table 9-1 for settings.

Table 9-1. Boot Mode Contact Settings

BMOD[1:0]	Boot Type
00	Internal Boot
01	Reserved
10	Internal Boot – ROM Select
11	Serial Downloader

9.4.1 Internal Boot (BMODE = 00)

Internal boot mode is selected by a value of ‘00’ on the BMOD[1:0] pins, at device power up. In this mode the processor boots from internal ROM. The boot code performs HW initialization, application image validation using the HAB library, and then jumps to an address derived from the application image. If any error occurs during internal boot the boot code jumps to the serial downloader. Internal boot mode is the only mode in which a secure boot of the i.MX51 is possible.

When set to Internal Boot (BMOD[1:0] = 00) the boot flow is controlled by a combination of eFUSE settings with the option of overriding the fuse setting using GPIO. The selection between these two boot modes is controlled by the GPIO Boot Select (GPIO_BT_SEL) fuse.

- If the GPIO_BT_SEL fuse is blown, all boot options are controlled by the eFUSEs described in Table 9-2 on page 9-5. The boot ROM software may read the value of BMOD[1:0] in theSBMR, or read the eFUSEs directly via the IIM module.
- If the GPIO_BT_SEL fuse is intact, the boot options are determined by the settings of the SBMR register. Some fuse options can be overridden in this mode. The fuses that can be overridden in GPIO mode are indicated by a YES in the GPIO column. See Table 9-4 on page 9-10 for details about the boot GPIO pins. In this mode the options’ values can only be read from the SBMR register.

The use of GPIO overrides is intended for development board work because those pads are used for other purposes in normal mode. Freescale recommends controlling the boot configuration by eFUSEs (GPIO_BT_SEL fuse blown) for deployed products and reserve the use of the GPIO mode (GPIO_BT_SEL fuse intact) for testing purposes. On production board GPIOs are not required—the customer can burn the fuses and not use the GPIO signals.

9.4.2 Internal Boot—ROM Select (BMODE = 10)

Internal boot (from only boot fuses) is selected by driving value of ‘10’ on the BOOT_MODE[1:0] pins, at device power up. This mode is equivalent to the Internal boot BOOT_MODE[1:0] = 00, with the only difference being that GPIO boot override pins are ignored, regardless of the BT_GPIO_SEL setting. The boot program only uses the boot eFUSE settings. This allows the user to burn fuses on the closed production device, with no external muxes on BOOT_MODE, pull ups/pull downs, and with no uncertainty that the serial downloader will be invoked by unknown boot pin values during the initial boot of the product device.

If set to Internal Boot ROM Select, the boot flow can be redirected if the BT_BLANK fuse is not blown (indicating that the ROM has not yet been programmed). This causes the boot flow to jump to the serial downloader. If the BT_BLANK fuse is blown the boot flow is normal and controlled by the eFUSE settings.

The first time a boad is used, when no fuses have been burnt yet, the device connected to BT GPIO pads can drive some values that will be incorrectly interpreted by ROM code. In such case the ROM code may try performing boot from not existing device. This may cause electrical/logic violation on some pads. Internal Mode—ROM Select (BMOD = 10) solves this problem. The first time the BT_BLANK fuse is encountered it is not burnt and therefore the ROM code jumps to the serial downloader. The next time BT_BLANK is burnt and therefore ROM code will perform internal boot according to the fuse settings.

The user can set BOOT_MODE[1:0] = 10 on a production device and burn fuses on the same device (by falling back to serial downloader), without changing value of BOOT_MODE[1:0] or pull ups/pull downs on the boot pins.

For cleaner jumping to serial downloader during initial fuse burning, the BT_BLANK fuse was introduced. If BOOT_MODE[1:0] = 10 and BT_BLANK = 0, then the ROM code jumps directly to the serial downloader, without trying other interfaces. BT_BLANK is designed to be blown by the user during initial fuse burning.

Table 9-2 shows the boot eFUSE descriptions.

Table 9-2. Boot eFUSE Descriptions

Fuse	Config	Definition	GPIO[1]	Settings[2]
DIR_BT_DIS	OEM	Direct External Memory Boot Disabled	NA	0 Direct boot to external memory is allowed 1 Direct boot to external memory is not allowed
BT_MEM_CTL[1:0]	OEM	Boot Memory Control Type used to select one of the following: • EIM (NOR, OneNAND) • NAND flash • Expansion devices such as SD/MMC or EEPROM.	Yes	00 WEIM 01 NAND Flash 10 Reserved 11 Expansion Device (SD/MMC, support high storage, EEPROMs. The fuse BT_MEM_TYPE[1:0] defines Expansion Device settings.

Table 9-2. Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Settings[2]
BT_PAGE_SIZE[1:0]	OEM	NAND Flash Page Size. This field is used in conjunction with the BT_MEM_CTL[1:0] setting to set page size of NAND flash device.[3] **Note:** BT_MEM_CTL must be set to NAND Flash for these settings to be used.	Yes	00 512 bytes 01 2 Kbytes 10 4 Kbytes 11 Reserved
BT_SPARE_SIZE	OEM	Specifies the size of spare bytes for 4Kbyte page size NAND Flash devices. **Note:** The spare size setting is only applicable when used with 4Kbyte page size devices. This fuse can also be used as a “fast boot” mode indication for use with eSD 2.10 protocol devices.	Yes	0 128 bytes spare (Samsung) 1 218 bytes spare (Micron, Toshiba) If the bootable device is SD then: 0 “FAST_BOOT” bit 29 in ACMD41 argument is 0 1 “FAST_BOOT” bit 29 in ACMD41 argument is 1
BT_BUS_WIDTH[1:0]	OEM	NAND/NOR Bus Width. **Note:** OneNAND devices can only use a 16-bit bus width,	Yes	BT_MEM_CTL[1:0] = NAND Flash 00 8-bit 01 16-bit BT_MEM_CTL[1:0] = WEIM (NOR) 00 16-bit data bus 01 32-bit data bus BT_MEM_CTL[1:0] = Expansion Device (SPI) 00 2-Address word SPI device (16-bit) 01 3-Address word SPI device (24-bit)
BT_MEM_TYPE[1:0]	OEM	Boot Memory Type. Selects memory card options. Available settings are defined by the memory type selected using **BT_MEM_CTL**	Yes	BT_MEM_CTL = 00 (WEIM) 00 NOR 01 Reserved 10 OneNand 11 Reserved BT_MEM_CTL = 01 (NAND Flash) 00 3 address cycles 01 4 address cycles 10 5 address cycles 11 reserved BT_MEM_CTL = 11 (Expansion Card) 00 SD/MMC/eMMC/eSD 01 Reserved 10 Serial ROM via I^2C 11 Serial ROM via SPI

Table 9-2. Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Settings[2]
BT_SRC[1:0]	OEM	Boot Source - This fuse selects one of the following Expansion card device types. • eSDHC 1/2/3/4 • I^2C1/2, HS-I^2C • CSPI, eCSPI1/2 **Note:** BT_MEM_CTL[1:0] must be set to 11 (Expansion Card Device) to use these settings	Yes	BT_MEM_TYPE = 00 (SD/MMC/eMMC/eSD) 00 eSDHC-1 01 eSDHC-2 10 eSDHC-3[4] 11 eSDHC-4[2] BT_MEM_TYPE = 10 (Serial ROM via I^2C) 00 I^2C-1 01 I^2C-2 10 HS-I^2C 11 Reserved BT_MEM_TYPE = 11 (Serial ROM via SPI) 00 eCSPI1 01 eCSPI2 10 CSPI 11 Reserved
BT_WEIM_MUXED[1:0]	OEM	Selects WEIM muxed mode.	Yes	BT_MEM_CTL[1:0]=00 (WEIM [NOR]) 00 Not muxed, not multiplexed with NAND, 16-bit data (high half) NOR interface. 01 Muxed, not multiplexed with NAND, 16-bit data (low half) NOR interface. 10 muxed, multiplexed with NAND data bus, 16-bit data (low half) NOR interface. 11 Muxed, multiplexed with NAND data bus, 32-bit data NOR interface.
BT_UART_SRC[1:0]	OEM	Selects the specific UART controller used for serial downloads.	Yes	00 UART1 01 UART2 10 UART3 11 Reserved
BT_MLC_SEL	OEM	SLC/MLC NAND device select or FAST BOOT enable/disabled for eMMC device	Yes	If BT_MEM_CTL[1:0] = NAND 0 SLC NAND device 1 MLC NAND device If all of the following conditions are met: • BT_MEM_CTL[1:0] = Expansion Device • BT_MEM_TYPE[1:0] = 00 • The bootable device is an MMC 0 eMMC fast boot mode disabled. 1 eMMC fast boot mode enabled
BT_EEPROM_CFG	OEM	Selects whether EEPROM device is used for loading configuration DCD data prior to booting from other devices (not applicable when using EEPROM as boot device)	Yes	0 Use EEPROM DCD 1 Do not use EEPROM DCD
BT_USB_SRC	OEM	USB PHY selection is based on this fuse setting.	Yes	0 USB-OTG internal UTMI PHY 1 USB-OTG external ULPI PHY

Table 9-2. Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Settings[2]
OSC_FREQ_SEL[1:0]	OEM	CKIH Frequency Select. It is used by boot code for PLL programming.	Yes	00 CKIH Frequency is 26 MHz 01 CKIH Frequency is 19.2 MHz 10 CKIH Frequency is 27 MHz 11 CKIH Frequency is 24 MHz
GPIO_BT_SEL	Freescale	GPIO Boot Select. Determines, whether the boot settings indicated by a Yes in the GPIO column are controlled by GPIO pins or eFUSE settings in the IIM: **Note:** This feature requires BMOD = 00 to operate.	NA	0 Select bits of SBMR are updated by GPIO. 1 Specific bits of SBMR are updated by IIM eFUSE settings.
HAB_TYPE[2:0]	OEM	Security Types as defined in Section 9.4 **Note:** In Engineering mode if CSF and SRK is not provided it must set to NULL in application header.	NA	001 Engineering (allows any code to be flashed and executed, even if it has no valid signature) 100 Security Disabled (For internal/testing use) OthersProduction (Security On)
SRK_HASH[255:0]	OEM	Most significant byte of 256-bit hash value of super root key (SRK_HASH)	NA	Settings vary –used by HAB
HAB_CUS[7:0]	OEM	HAB Customer Code — Selects customer code, as input to HAB.	NA	Settings vary –used by HAB
DIE-X-CORDINATE[7:0] DIE-Y-CORDINATE[7:0] WAFER_NO[4:0] LOT_NO_ENC[42:40] LOT_NO_ENC[39:32] LOT_NO_ENC[31:24] LOT_NO_ENC[23:16] LOT_NO_ENC[15:8] LOT_NO_ENC[7:0]	Freescale	Device Unique ID, 64-bit UID.	NA	Settings vary –used by HAB
SRTC_SECMODE[1:0]	OEM	Security Mode for Secure RTC. Determines the level of security of the Secure Real Time Clock (SRTC) module	No	00 Low Security 01 Medium Security 10 High Security 11 Reserved
BT_LPB_FREQ[2:0]	OEM	Low-Power Boot Mode (LPBM) ARM core frequency.	Yes	000 192 MHz (Default - out of reset), 001 133 Mhz 010 55.33 MHz 011 200 MHz 100 220 MHz 101 166 MHz 110 266 MHz 111 Normal boot frequency (400 MHz)
BT_BLANK	OEM	Indicates that the boot area has not yet been burned. **Note:** This fuse is only read when BOOT_MODE[1:0]=10	No	0 BOOT area is unprogrammed. Boot flow jumps to serial downloader. 1 BOOT area is programmed. Regular boot flow is performed.

Table 9-2. Boot eFUSE Descriptions (continued)

Fuse	Config	Definition	GPIO[1]	Settings[2]
MMU_EN	OEM	MMU/d-cache enable bit used by boot ROM for fast HAB processing	No	0 MMU/d-cache is disabled by ROM during the boot 1 MMU/d-cache is enabled by ROM during the boot
BT_LPB[1:0]	OEM	Options for Low-Power Boot Mode (LPBM).	No	00 LPBM disabled 01 Generic PMIC and one GPIO input (Low battery) 10 Generic PMIC and two GPIO inputs (Low battery and Charger detect) 11 Atlas AP Power Management IC.

1 Setting can be overridden by GPIO settings when GPIO_BT_SEL fuse is intact. See Table 9-4 for corresponding GPIO pin.

2 0 = intact fuse and 1= blown fuse

3 The Page-Per-Block settings may not always exhibit a “one to one” relationship to the block size; however, most devices exhibit the relationships listed in Table 9-3:

4 eMMC4.3 fastboot is supported

Table 9-3. Page-Per-Block Relationships

Page Size	Page-per-Block
512 Bytes	32
2 Kbytes SLC	64
2 Kbytes MLC	128
4 Kbytes	128

9.4.2.1 GPIO Boot Overrides

Table 9-4 on page 9-10 provides a listing of GPIO boot overrides. These input pins are sampled at boot and can be used to override corresponding fuse values, depending on the setting of the GPIO_BT_SEL fuse. The boot GPIO override options are only in effect when GPIO_BT_SEL is ‘0’ (intact fuse) and BOOTM[1:] = 10.

Table 9-4. GPIO Override Contact Assignments

Contact	eFUSE	Details
DISP1_DAT[21:20]	BT_MEM_TYPE[1:0]	GPIO Boot pin overrides fuse settings if the following conditions apply: • Internal Boot mode (BMODE[0:1]=00) • GPIO_BT_SEL = 0
DISP1_DAT[19:18]	BT_WEIM_MUXED[1:0]	
DISP1_DAT[17:16]	BT_PAGE_SIZE[1:0]	
DISP1_DAT[15]	BT_BUS_WIDTH	
DISP1_DAT[14:13]	BT_MEM_CTL[1:0]	
DISP1_DAT[12]	BT_MLC_SEL	
DISP1_DAT[11,23,22]	BT_LPB_FREQ[2:0]	
DISP1_DAT[10]	BT_SPARE_SIZE	
DISP1_DAT[9:8]	BT_SRC[1:0]	
DISP1_DAT[7]	BT_EEPROM_CFG	
DISP1_DAT[6]	BT_USB_SRC	
EIM_A[23]	BT_HPN_EN	
EIM_A[21:20]	BT_UART_SRC[1:0]	
EIM_A[19:18]	BT_LPB[1:0]	
EIM_A[17:16]	OSC_FREQ_SEL[1:0]	

9.4.3 Serial Downloader (BMOD[1:0] = 11)

The serial downloader is invoked if the external Flash device is not programmed, when a failure is encountered during the boot flow process or any of the following conditions are met:

- BMOD[1:0] = 11 (serial downloader mode)
- BMOD[1:0] = 10 (internal boot with fuses) and the eFUSE BT_BLANK = 0
- BMOD[1:0] = 10 (internal boot with fuses) but the fuses are not set properly.
- BMOD[1:0] = 00 or 10 (internal or internal boot with fuses) and there is not a valid image in the Flash device
- Security hardware failure
- Runtime exception occurs
- Error returned by the HAB functions while in production mode. Errors are ignored in engineering mode)

To determine the active serial port, either UART or USB, the processor ROM program polls the UART and USB status register for approximately 32 seconds. If there is no activity on either port within the predefined polling time, the ROM program powers down the IC using WDOG. When the serial downloader is active, the WDOG is serviced periodically. If the communication between the serial host

and the i.MX51 hangs for more than 32 seconds or the processor enters an endless loop, the WDOG expires and powers down the device.

NOTE

For detailed information about the boot module contact your Freescale representative.

Chapter 10
Multimedia

10.1 Video Subsystem

The video subsystem includes the following dedicated modules:

- Video Processing Unit (VPU): a multi-standard video
- Image Processing Unit (IPU): providing connectivity to displays, related processing, synchronization, and control
- TV encoder (TVE) bridge: providing optional translation from the digital display interface supported by the IPU to SDTV analog and some HDTV interfaces

These modules are connected to the other parts of the system as follows:

- IPU has two display ports, used to connect to relevant external devices.
- The VPU and IPU have a master AXI port, providing access to system memory. The IPU provides different IDs for accesses related to real-time and non-real-time flows (real-time flows being screen refresh) to allow allocating a higher priority to the real-time flows, reducing the risk of their starvation.
- All modules have a host interface, used to configure and control them (by the ARM MCU or the SDMA). For the VPU and the TVE, this is a slave IP port. For the IPU, this is a slave AHB port.
- The slave AHB port of the IPU also provides direct access from the ARM MCU and SDMA to an external display controller or graphics accelerator, connected to the display port.

10.1.1 Image Processing Unit (IPU)

Table 10-1 shows the IPU IP parametric table.

Table 10-1. IPU IP Parametric Table

Name	IPU
Function	Connectivity to displays; related processing; synchronization and control
External I/O Pins Notes: This is the pinout of the IPU module At chip level, some of the pins are muxed and some are omitted. Additional GPIO pins are required to construct the connection. This is not included in this list.	Parallel Display port: 32 bit data, ~18 clocks and controls. Regular CMOS IO type, 133 MHz max. May be also connected to the TVE internal connectivity bridge.

Table 10-1. IPU IP Parametric Table (continued)

Name	IPU
SoC Buses	AXI master—for accessing the memory AHB slave—for programming, control and direct access of the MCU to the display
Interrupts	Two interrupts: functional and error
DMA Requests	Includes an integral DMA controller, with an AXI master port Also one DMA request to the SDMA
Number of instantiations	1
Clock sources and range	HSP_CLK—Internal high-speed processing clock: up to 133 MHz DI_CLK—Display interface clock: up to 80 MHz

The goal of the IPU is to provide comprehensive support for the flow of data to a display device. This support covers all aspects of these activities:

- Connectivity to relevant devices—displays, graphics accelerators, TV encoders
- Related image processing and manipulation: image enhancements and conversions, etc.
- Synchronization and control capabilities (for example, to avoid tearing artifacts)

This integrative approach leads to several significant advantages:

- Automation
  The involvement of the MCU (Main Control Unit) in image management is minimized. In particular, display refresh/update can be performed completely autonomously. The resulting benefits are reducing the overhead due to software-hardware synchronization, freeing the MCU to perform other tasks and reduced power consumption (when the MCU is idle and can be powered down).
- Optimal data path
  Access to system memory is minimized. In particular, significant processing can be performed on-the-fly while sending data to a display. System memory is used essentially only when a change in pixel order or frame rate is needed. The resulting benefits are reduced load on the system bus and further reduction of power consumption.
- Resource sharing
  Maximal hardware reuse for different applications, resulting with the support of a wide range of requirements with minimal hardware

The hardware reuse mentioned above is enabled by a sophisticated configurability of each hardware block. This configurability also allows the support of a wide range of external devices, data formats and operation modes. The resulting flexibility is important because the support requirements are evolving significantly; expected future changes need to be anticipated and accounted for.

10.1.1.1 External Ports

The IPU has the following ports:

- Two display ports—each controlled by a DI module—providing a connection to displays and related devices.
- Memory port—AXI (AHB V3.0) master, controlled by the IDMAC—providing connection to the system memory.
- AHB-lite slave port, providing connection to the ARM MCU (and to any other master connected to the ARM's cross-bar switch)
- Additional ports for control and debug

10.1.1.1.1 Display Ports

The role of these ports is to communicate with a display device, either directly or through a controller (for example, graphics accelerator) or a bridge (for example, TV encoder).

Two access modes are supported: synchronous access and asynchronous access.

In synchronous access mode, the IPU transfers a two-dimensional block of pixels to the display device, in synchronization with the screen refresh cycle.

This mode has a dual role:

- For a RAM-less display or a TV screen, this mode is used to perform the screen refresh process from a display buffer in system memory.
- For a "smart" display, this mode is used to transfer a rectangular block of pixels to the display's screen and, in some cases, also to the display buffer

In both cases, the IPU sends to the display all the synchronization signals controlling the screen refresh and the block transfer is synchronized with these signals. This synchronization means that tearing effects are avoided when using this mode.

Asynchronous access is the main mode used for communicating with an external display controller (possibly in a smart display or a graphics accelerator). In this mode, the IPU performs random access—read/write—to the memory and registers of controller.

The following access types are provided:

- Data transfer to the external device, after on-the-fly processing in the IPU.
- Data transfer (DMA)—read/write—between the host's system memory and the external device, through the IPU's memory port (controlled by the IDMAC); for example, transfer of a rectangular block of pixels (possibly full screen).
- Host access—read/write—to an external device, through the AHB-slave port
  - Access types
    - Direct access—emulating a directly-addressed access (see below)
      This includes burst access (incremental; up to 8 words/burst)
    - Low-level access—leaving to the host the explicit generation of the access protocol

— The possible accessing modules include the MCU and the system DMA controller (as well as any other AHB master connected to the MCU's cross-bar switch).

Asynchronous access requires the specification of an address. The display interface uses indirect addressing, meaning there is no address bus and the address, control commands, and configuration commands, are embedded in the data stream. The access procedure, including writing addresses and commands, can be managed autonomously by the interface, using an access template generated by the MCU.

10.1.1.1.2 The Interface

The display interface is very flexible and supports a wide variety of devices from major manufacturers. The following interface types are provided (in each of the two display ports).

- A parallel video interface (for synchronous access)—up to 24-bit data bus.
- A parallel bidirectional bus interface (for asynchronous access)—up to 32-bit data bus.

The supported formats for pixel data are: RGB and YUV 4:2:2 (for TV encoder).

The transfer rate supported is at least 100 MHz (for all interfaces)

- For synchronous access with one cycle/pixel, this enables, e.g (including 35% blocking intervals)
  - XGA (1024 × 768) at 100 fps
  - 720p (1280 × 720) at 60 fps
  - 1080i (1920 × 1080) at 30 fps
- When two (or more) display devices are used simultaneously, the total rate supported is as above.

Simultaneous functionality of the above devices is possible in each of the following ways:

- Two devices can be accessed independently, each through a different port.
- Two devices can time-share asynchronous accesses, using the CS signals.
- An asynchronous access can be performed during vertical blanking intervals of a synchronous access.

10.1.1.2 Processing

The IPU processes rectangular blocks of pixels. The processing is performed in modules: DP, IC, and IRT. Several time-shared data flows are supported, as described in Table 10-2.

Table 10-2. Time-Shared Data Flows Through The IPU

Name	Number	Type	Flow	Target	Restrictions
Display Refresh/ Update	5 flows (at most two of them of type DS1)	DS1	Fmem → DP → Display	Synchronous Access (for example, display refresh; controlled by the DI)	—
		DS2	Fmem → DP → Display	Asynchronous Access (for example, display update)	—
		DS3	Fmem ↔ Display	Generic Data Transfer	—
	1 flow	DS4	MCU ↔ Display	Direct Access	—

Table 10-2. Time-Shared Data Flows Through The IPU (continued)

Name	Number	Type	Flow	Target	Restrictions
Video Playback	flow	PL1	Bmem → IC → Bmem → IRT → Fmem + DSx	Main option	—
		PL2	Fmem → IRT→ Bmem → IC → DP	Low power (branching to DSx, as a video plane)	Large enough window No other video flows
Graphic Overlays	2 flows	GF1	Fmem → IC	(combining with the main flow)	—
	2 flows	GF2	Fmem → DP		—

Comments

- System memory usage–legend
  - — Fmem: frame double-buffer (page-flip) in system memory (typically external)
  - — Bmem: two possibilities
    - – A frame double buffer, as above
    - – A band (4–256 rows) double-buffer (page-flip) in system memory (could be internal)
  - — Direct arrow between two processing stages represents an internal pipelining
- Time-sharing
  - — DP can time-share one DS1 flow and a one DS2 flow (each with different destinations and independent processing parameters)
  - — Direct access to display (DS4) time-shares tightly the display port with other active DSx flows.
  - — Other time-sharing (between PLx, between DS2 and DS3, in IRT) is frame-by-frame

Any of the processing stages in the above flows can be skipped.

10.1.1.2.1 Display Processor (DP)

The display processor performs processing required for output to a display:

- Combining two video/graphics planes
- Overlaying a simple hardware cursor
  32×32 pixels, uniform color, may be combined logically with the background.
- Color conversion/correction—linear (multiplicative and additive)
  Programmable; including:
  - — YUV ↔ RGB, YUV ↔ YUV conversions
    where YUV stands for any one of the color formats defined in the MPEG-4 standard
  - — Adjustments: brightness, contrast, color saturation...
  - — Special effects: gray-scale, color inversion, sepia, blue-tone...
  - — Hue-preserving clipping, for gamut mapping
  - — Applied to the output of combining or to one of the inputs
- Gamma correction and contrast stretching—programmable piecewise-linear map

The DP processes a single data flow at any given time, but it supports up to three data flows, by time sharing, one of them may be synchronous.

The data throughput is up to 110M pixels/sec (peak; including blanking intervals)

10.1.1.2.2 Image Converter (IC)

The Image Converter performs various operations on a video stream. The operations performed are:

- Resizing
  - Fully flexible resizing ratio
    Maximal downsizing ratio: 8:1
    Subject to this limitation, any N→M resizing can be performed
  - Independent horizontal and vertical resizing ratios.
- Color conversion/correction - linear (multiplicative and additive)
  Programmable; including:
  - YUV ↔ RGB, YUV ↔ YUV conversions
    where YUV stands for any one of the color formats defined in the MPEG-4 standard
  - Adjustments: brightness, contrast, color saturation...
  - Special effects: gray-scale, color inversion, sepia, blue-tone
- Combining with a graphics plane (for example, application-specific overlay)
- Horizontal inversion

The IC supports three time-shared data flows share a common input).

The frame resolution supported is up to 4096 × 4096 for input and up to 1024 × 1024 for output. Wider frames can be processed by the IC by splitting them to vertical stripes.

The data throughput is up to 100 M pixels/sec for input and up to 50 M pixels/sec for output.

10.1.1.2.3 Image Rotator (IRT)

The Image Rotator performs any combination of the following:

- 90-degree rotation
- Horizontal inversion
- Vertical inversion

The data throughput is up to 50 M pixels/sec.

10.1.1.3 Automatic Procedures

The IPU is equipped with powerful control and synchronization capabilities to perform its tasks with minimal involvement of ARM and minimal use of memory. In particular, it includes the following:

- An integrated DMA controller with an AXI master port, allowing autonomous access to the system memory
- An integrated display controller, performing screen refresh of a RAM-less display.

- A page-flip double buffering mechanism, synchronizing read and write access to system memory, to prevent tearing effects.
- Internal synchronization

As a result, in most cases, the MCU is involved only when it also performs part of the processing (for example, video coding). In particular, the following procedures are performed by the IPU completely autonomously:

- Screen refresh for RAM-less displays
- Update of the (foreground) display buffer used for screen refresh (located either in system memory or in an external display controller, for example, of a smart display or graphics accelerator) when the content is generated in a different (background) buffer.

Typically, there are extended periods of time in which there is no other activity in the system. The MCU being idle can be put into a low-power mode, reducing the power consumption and significantly extending the battery life.

The IPU supports the following techniques that further reduce the power consumption of the display system:

- Optimized update of the display buffer, using a snooping signal from the ISM (IPU Snooping Module), indicating a modification of the source buffer.
- Dynamic backlight control, with low-light compensation by image enhancement (contrast and brightness adjustment)

Further features and capabilities of the automatic procedures include the following:

- Automatic display of a changing image (animation) or moving image (scrolling).
- The timing of the display update can be adjusted to avoid tearing.

The IPUE supports direct frame synchronization with the GPU, using either two or three frames, as follows:

- The GPU instructs the IPU which frame should be transferred to the display.
- The IPU notifies the GPU which frame is currently transferred to the display.
- The IPU provides to the GPU an end-of-frame trigger (after which it switches to the frame indicated by the GPU).

This mechanism is provided for the main plane of the primary flow to the DP.

The clock sources received by the IPU are listed in Table 10-3.

Table 10-3. IPU Clock Sources

Name	Symbol	Source	Rate	Comments
High-Speed Processing Clock	HSP_CLK	Clock control Module	Up to 133 MHz	—
Display Interface Clock	DI_CLK	Clock control Module or an external PLL	up to 80 MHz	Optional For example, for synchronization with a TV encoder

10.2 Video Processing Unit (VPU)

The VPU is a multistandard video codec (encoder/decoder) capable of handling up to four simultaneous multiple streams using time multiplexing. The VPU is a very flexible block consisting of hardwired accelerators surrounding a programmable core. The VPU presents to system a register mapped interface that is controlled by the embedded processor. Because this interface can be updated by the firmware, it is not documented in this document. Instead, the designer should consult the documentation released with the firmware used. End users should only interface with the VPU using the API that is also released with each firmware release. This API isolates the user from possible changes in the register level interface.

Table 10-4 shows a summary of the VPU specs. The VPU has its own DMA driven AXI masters that allow it to retrieve the required data directly from system memory. The load in the host is negligible because it only needs to interact with the VPU at the frame level.

Table 10-4. A Brief Summary of VPU Specification

Name	VPU
Function	Decode video streams including optional video processing such as rotation, deringing and mirroring.
Supported encoders	MPEG-4 SP H.263 V2 + Annex J, K (RS = 0 and ASO = 0), and T H.264 BP MJPEG Baseline
Supported decoders	MPEG-2 MP VC-1 SP, MP, HP MPEG-4 SP, ASP H.263 V2 + Annex J, K (RS = 0 and ASO = 0), and T H.264 BP, MP, HP DivX v3,4,5 Real Video 10 MJPEG Baseline
External I/O Pins (List, Type, Schmidt Trigger, Speed)	No external I/O pins are needed
SoC Buses (List, Type, Bandwidth)	64 bit AXI master for accessing the system memory and search RAM IPBus slave for host control
Interrupts	one interrupt
DMA Requests	Integrated DMA controller on the AXI master port
Endianness	64 and 32 bit BE/LE
Number of instantiations	1
Clock sources and range	Core clock: up to 133 MHz AXI bus clock: up to 166 MHz IP bus clock: up to 66.5 MHz

Chapter 11
Power Management

The i.MX51 supports several power management techniques to reduce active and static power consumption. This chapter describes the operation of these features and the registers used to configure and control them.

11.1 Power Saving Methodology

This section discusses active power savings and leakage power savings.

11.1.1 Active Power Savings

Active power saving methods include the following:

- Dynamic voltage frequency scaling (DVFS)

  This reduces active power consumption by scaling voltage and frequency. There are two DVFS engines in the i.MX51: one is used by the peripherals and the other by the ARM platform.
- Dynamic process temperature compensation (DPTC)

  This reduces active power consumption by adjusting supply voltage according to the individual device's characteristics and ambient temperature. There are two DPTC engines: one is used by the peripherals and the other by the ARM platform.
- Clock gating

  This reduces active power consumption by gating the clock to each module while the module is in idle state. The i.MX51 implements three levels of clock gating as follows:
  - Clock tree roots in the Clock Control Module (CCM)
  - Clock tree branches in the Low Power Clock Gating unit (LPCG)
  - Clock tree leaf nodes in the individual modules

11.1.2 Controlling Leakage

The primary leakage control mechanisms are reduced voltage (STOP mode) and power gating. There are three types of power gating, as follows:

- State retention power gating (SRPG)
- Power gating (PG)
- PG with Save and Restore

The three types of power gating work in the following ways. When SRPG is applied, the state of the flip flops is preserved to internal switch cells (powered by a continuously powered supply) at the time the combinatorial logic between the flip flops is powered down. Using PG removes power from the selected

module, resulting in the loss of all data in three affected modules: the VPU, GPU2D, and GPU3D. When PG with Save and Restore is used, the state of the block is saved into its constantly powered memory before the power to the block is removed. Once power is reapplied, the state of the block is reloaded from memory, and the block continues in the same state as it was in before it was powered down. This type of power gating is accomplished by hardware in the IC. IPUEX supports PG with Save and Restore.

11.2 Power Gating Sequences

The following order is recommended when powering up the modules in the i.MX51:

1. SoG—After the SoG has powered up, move to the next power-up stage.
2. IPUEX—After the IPUEX has powered up, move to the next power-up stage.
3. VPU—After the VPU has powered up, move to the next power-up stage.
4. GPU3D—After the GPU3D has powered up, move to the next power-up stage.
5. GPU2D—After the GPU2D has powered up, move to the next power-up stage.
6. EMI—After the EMI has powered up, an acknowledgement the power-up is complete is generated.

11.2.1 Power Gating Options

The sequence of power gating is controlled by the Global Power Controller (GPC) modules on the i.MX51. The following power gating options exist:

- SRPG of the Cortex_A8 platform in WAIT mode.
- SRPG of the Cortex_A8 platform in STOP mode.
- SRPG of the NEON floating unit when it’s not needed, by software control.
- Partial SRPG of the EMI block during WAIT mode.
- Partial SRPG of the EMI block during STOP mode.
- Power gating of one or more of the following blocks during WAIT mode: VPU, GPU3D and GPU2D.
- Power gating of one or more of the following blocks during STOP mode: VPU, GPU3D and GPU2D.
- Power gating of one or more of the following blocks during RUN mode: VPU, GPU3D and GPU2D.
- Power gating with PG with Save and Restore of IPUEX during WAIT.
- Power gating with PG with Save and Restore of IPUEX during STOP.

11.3 Low-Power Modes

The i.MX51 can operate in several different low-power modes. Some low-power modes are module specific. The four low-power modes as follows:

- RUN

The ARM core is active; its clocks are on; and the peripheral modules needed for a specific task are active. Software is used to gate off clocks from modules that are not in use. The CCM can enable power gating of the modules described above.

- WAIT

  The ARM core is disabled; its clocks are gated off; and the bus clocks to peripherals are gated on as required. The power gating modes PG and SRPG can be applied to Cortex_A8 and the different blocks as described in Chapter 7, "Clock Controller Module (CCM)," Section 11.4, Module Specific Power Modes."

- STOP

  The ARM core is disabled; peripherals are disabled; bus clocks are off; and clock PLLs are off. The power gating modes PG and SRPG can be applied to Cortex_A8 and the different blocks as described in Chapter 7, "Clock Controller Module (CCM)," Section 11.4, Module Specific Power Modes."

- LPSR—Low Power Screen Refresh

  This is a subset of the STOP mode. The ARM core is disabled; all peripherals but the ones needed for LPSR are disabled; and clock PLLs are off. PG and SRPG can be applied to the Cortex_A8 and the peripherals that are not needed for LPSR. In this mode, the clocks are supplied by either FPM or CKIH. The peripherals that are required for LPSR mode are as follows:

  — IPU EX

  — EMI—only the internal memory arbitration is needed.

  — SCC and its internal memory

  — All translators between the EMI to the SCC.

11.4 Module Specific Power Modes

The power modes of the individual modules are as follows:

- Active—Operational with clocks on.
- Idle—Not operational but the clocks are on.
- Disabled—Not operational and the clocks are off.

The module modes can be mapped onto the domain modes as shown in Table 11-1

Table 11-1. Low-Power Modes

Mode	ARM Platform	Modules	PLL	CKIH/FPM	CKIL
RUN	Active	Active, Idle, or Disabled[1]	On	On	On
WAIT	Disabled	Active, Idle, or Disabled[1]	On	On	On
STOP	Disabled	Disabled[2]	Off	Off	On
LPSR	Disabled	Disabled[3]	Off	On	On[4]

1 During RUN and WAIT modes, the peripherals can be active, but not all must be active. The modules that are not needed can be Idle or Disabled in the RUN and WAIT modes.

2 Some modules can operate using CKIL in STOP mode.

[3] Some blocks need to operate using CKIH or FPM in LPSR mode.

[4] If the FPM is selected then CKIL needs to be left on.

11.4.1 Power Down Sequence

The power-down sequence is as follows:

1. The software configures the LPM bits in the CCM (see CCM chapter) as well as the power gate bits in the GPC. The information (if a peripheral or the ARM platform can be power gated) is stored in the GPC module.
2. The ARM platform starts the LPM sequence, resulting in the CCM raising a power-down request for ARM and the peripherals.
3. The power-down sequence occurs in parallel on all peripherals and the ARM platform.
4. A combined power-down acknowledge signal is issued to the CCM from the GPC module.

11.4.2 Power Up Sequence

The power-up sequence is as follows:

1. After an interrupt is received and the clocks resume operation, a power-up request signal is issued from the CCM to the GPC.
2. The power-up of the ARM core and the other peripherals occur in parallel.
3. The power-up of the peripherals is done one after the other in order to prevent excessive current drain during the power up.
4. The power-up request of the CCM module is connected to the power-up request of the first PGC block and its power-up acknowledge is connected to the power-up request of the next PGC blocks.
5. Each of the PGC modules sends a reset request to the SRC module. This initiates the reset sequence to the needed module as described in the SRC chapter.

11.4.3 IPU Save and Restore Sequence

The IPU save and restore sequence is as follows:

1. If the power gating bit of the IPU EX is selected, a signal (gpc_ipu_stat_pg) stating that IPU EX will be power gated in the next LPM is sent to the IPU EX.
2. In the process of entering low power mode, the IPU EX needs to approve the entrance to LPM (refer to the CCM chapter). If it sees the gpc_ipu_stat_pg, it goes through the process of Save in the IPU EX memories so that their arrays will always be powered. Only after the Save is complete does it return an acknowledge of the power-down to the CCM. The CCM then continues the LPM sequence.
3. In the process of exiting low-power mode, the IPU EX checks to see if the gpc_ipu_pg_event signal is active (meaning a power gate has occurred). Then the IPU EX restores all the required data from the memories.
4. The IPU EX sends a signal called ipu_stby_ack indicating it has completed the process. This signal clears the gpc_ipu_pg_event signal in the GPC.

11.5 RAM Memory Supplies

Refer to the Operating Ranges table in the i.MX51 data sheet.

11.6 Fusebox Supplies

Refer to the Operating Ranges table in the i.MX51 data sheet.

11.7 Dynamic Voltage and Frequency Scaling

The i.MX51 uses DVFS as one of its active power saving methodologies. The ARM core (VDDGP) and peripheral core (VCC) supply voltage ranges are stated in the Operating Ranges table of the i.MX51 data sheet.

DVFS is supported on both the ARM platform and the peripherals using the following two different load monitors:

- DVFS_Core
  - Monitors by weighing signals relevant to the ARM platform
  - Affects only the ARM platform.
- DVFS_Per
  - Monitors by weighing signal relevant to the i.MX51 peripherals
  - Affects only the peripherals.
  - While in DVFS mode each clock is divided by 2, 3, or 4 across all of the peripheral domain, depending on the configuration in the CCM.

Chapter 12
System Security

12.1 Introduction

Security is an increasingly important feature of handheld devices such as cell phones, ultra-portable computers, and integrated media players. Instances of hackers and pirates breaking into portable devices and stealing private information or copyrighted content are becoming more and more common. As such, security is a high priority for the i.MX51.

To address the potential security risks and to provide an extensible platform for addressing future security needs, the i.MX51 incorporates the following advanced hardware blocks and architectural features:

- High Assurance Boot (HAB) System
- Memory Management Unit (MMU)
- Trusted Execution Environment (ARM's TrustZone, Virtualization)
- TrustZone Interrupt Controller (TZIC) and TrustZone Watchdog (TZ WDOG)
- EMI WaterMark (WM) mechanism
- Central Security Unit (CSU)
- IC Identification Module (IIM) with On-chip One Time Programmable Electrical Fuse Array
- Security Controller (SCC) with 128KByte of secure RAM
- Cryptographic Accelerator (SAHARA4LT)
- Run-Time Integrity Checker (RTIC)
- Secure JTAG Controller (SJC)
- Secure Real Time Clock (SRTC)
- Physical Tamper Detectors
- Security Services and Protocols

For detailed information about the operation of the security features, contact your Freescale representative.

MCIMX51 Reference Manual
Book II

Rev. 1
2/2010

Chapter 13
1-Wire Module (1-Wire)

13.1 Overview

The 1-Wire module provides the communication link to a generic 1-Kbit add-only memory. The module sends or receives one bit at a time. The required protocol for accessing the generic 1-Wire device is defined by Maxim. The generic 1-Wire device holds battery characteristics information.

Figure 13-2 shows a block diagram of the 1-Wire module.

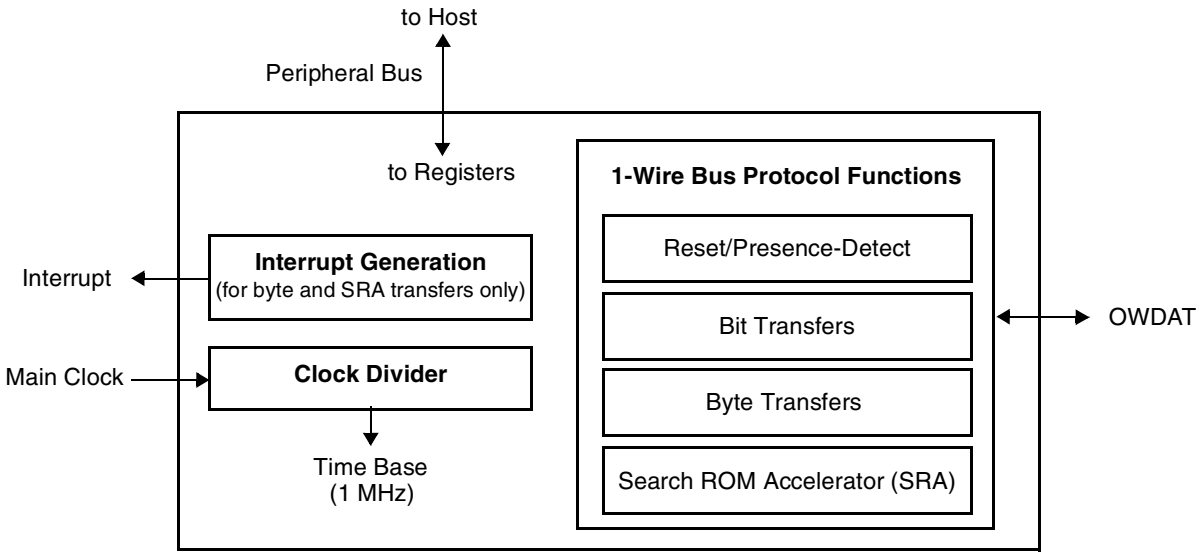


Figure 13-1. 1-Wire Module Block Diagram

13.1.1 Features

The 1-Wire module includes the following features:

- Performs the 1-Wire bus protocol to communicate with an external 1-Wire device.
- Provides a clock divider to generate a 1-Wire bus reference clock (derived from the main clock provided internally to the module).

13.1.2 Modes of Operation

The 1-Wire module supports the following operations:

- Normal Operating Modes (See Section 13.4.1, Normal Operating Modes")
  - — Bit Transfers
  - — Reset/Presence-detect Pulse
- Low-Power Mode (See Section 13.4.2, Low Power Mode")

13.2 External Signals

Table 13-1 shows the signal that interfaces with a generic 1-Wire device.

Table 13-1. 1-Wire Module Signal

Signal	I/O	Function
OWDAT	I/O	1-Wire bus Requires an external pull-up resistor. The recommended resistor value is specified by the generic 1-Wire device used in a given system.

13.3 Memory Map and Register Definition

This section provides the module memory map and detailed descriptions of all registers.

13.3.1 Memory Map

Table 13-2 shows the 1-Wire memory map.

Table 13-2. 1-Wire Memory Map

Base Address	Register	Access	Reset Value	Section/Page
0x83FA_4000 (CONTROL)	Control register	R/W	0x0000	13.3.2.1/13-2
0x83FA_4002 (TIME_DIVIDER)	Time Divider register	R/W	0x0000	13.3.2.2/13-4
0x83FA_4004 (RESET)	Reset register	R/W	0x0000	13.3.2.3/13-4

13.3.2 Register Descriptions

This section provides the detailed descriptions for the registers. All registers are byte-addressable.

13.3.2.1 Control Register (CONTROL)

The control register is used to initiate the reset/presence-detect sequence and bit transfers. The register also provides the presence-detect status and bit-read status.

Figure 13-2 shows the register. Table 13-3 describes the register fields.

Address 0x83FA_4000 (CONTROL) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	RPP	PST	WR0	WR1	RDST	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-2. Control Register

Table 13-3. Control Register Field Descriptions

Field	Description
15–8	Reserved
7 RPP	Reset/Presence-detect Pulse. This bit is self-clearing and is cleared after the presence or absence of an external device is determined. See Section 13.4.1.1, Reset/Presence-detect Pulse." When writing: 0 Do nothing. 1 Generate Reset Pulse and sample the bus for the presence pulse from the external device. When reading: 0 Reset pulse complete. 1 Sequence not complete.
6 PST	Presence Status. This bit is valid after the RPP bit is self-cleared. 0 Device is not present. 1 Device is present.
5 WR0	Write 0. This bit is self-clearing and is cleared when the write of the bit is complete. See Section 13.4.1.2.1, Write-0 Sequence." When writing: 0 Do nothing. 1 Write a 0 bit to the interface. When reading: 0 Write sequence complete. 1 Sequence not complete.
4 WR1	Write 1/Read. This bit is self-clearing and is cleared when the write sequence is complete. See Section 13.4.1.2.2, Write-1/Read Sequence." When writing: 0 Do nothing 1 Write a 1 bit to the interface and sample the bus. When reading: 0 Sequence complete. 1 Sequence not complete.
3 RDST	Read Status. This bit is valid after the WR1 bit is self cleared. 0 A 0 has been sampled. 1 A 1 has been sampled.
2–0	Reserved

13.3.2.2 Time Divider Register (TIME_DIVIDER)

The time divider register is used for dividing the main clock (ipg_clk) input down to 1 MHz to generate the module's time base.

Figure 13-3 shows the register. Table 13-4 describes the register fields.

Address 0x83FA_4002 (TIME_DIVIDER) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DVDR							
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-3. Time Divider Register

Table 13-4. Time Divider Register Field Descriptions

Field	Description
15–8	Reserved
7–0 DVDR	Divider Factor. The internal clock divider uses this field to generate the required time base for the module. See Section 13.4.3, Clocks." 0x00 1 (default) 0x01 2 … … 0xFF 256

13.3.2.3 Reset Register (RESET)

The reset register is used to perform a software reset of the 1-Wire module. Figure 13-4 shows the register. Table 13-5 describes the register fields.

Address 0x83FA_4004 (RESET) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RST
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-4. Reset Register

Table 13-5. Reset Register Field Descriptions

Field	Description
15–1	Reserved
0 RST	Software Reset. See Section 13.4.4.2, Software Reset." 0 Do not perform a software reset. 1 Initiate a software reset and hold the module in the software-reset state.

13.4 Functional Description

The 1-Wire module interfaces with a generic 1-Kbit add-only memory, through a simple 1-bit bus. Software uses the 1-Wire bus to program and read the 1-Kbyte memory.

The protocol involves first issuing one of four ROM function commands before the EPROM is accessible:

- Read ROM
- Match ROM

Through the 1-Wire bus, the host software interfaces with the generic 1-Wire device and allows the required commands to be issued to control the EPROM of a generic 1-Wire device. The host (through the 1-Wire interface) is the bus master, and the generic 1-Wire device(s) are the slave(s).

13.4.1 Normal Operating Modes

The 1-Wire module supports the following 1-Wire bus protocol functions:

- Reset/Presence-detect pulse using the control register (See Section 13.4.1.1, Reset/Presence-detect Pulse")
- Bit Transfers using the control register (See Section 13.4.1.2, Bit Transfers")

13.4.1.1 Reset/Presence-detect Pulse

The 1-Wire module provides for an automated initialization sequence for the 1-Wire bus. Software initiates the initialization sequence by setting CONTROL[RPP]. The automated initialization sequence is as follows:

1. Generate a reset pulse.
2. Listen for a response from an external device by sampling for the 1-Wire device presence bit.
3. After an amount of time determined by the 1-Wire standard, latch the presence bit (true or false) in CONTROL[PST].

If an external device is detected (PST = 1), software can begin communications on the 1-Wire bus.

The presence pulse is used by the 1-Wire to determine if at least one generic 1-Wire device is connected. Software determines if more than one generic 1-Wire device exists.

13.4.1.2 Bit Transfers

After the initialization sequence (see Section 13.4.1.1, Reset/Presence-detect Pulse"), software can write and read one bit at a time using the control register.

13.4.1.2.1 Write-0 Sequence

The Write-0 sequence writes a zero bit to the generic 1-Wire device. Setting the CONTROL[WR0] initiates the Write-0 pulse sequence. Once the write is complete, the WR0 bit is automatically cleared.

13.4.1.2.2 Write-1/Read Sequence

The Write-1 sequence writes a one bit to the generic 1-Wire device. Setting the CONTROL[WR1] bit initiates the Write-1 pulse sequence. Once the write is complete, the WR1 bit is automatically cleared.

Because the Write-1 and Read timings are identical, this sequence also reads a bit from the bus. The sampled value is stored in the read status bit CONTROL[RDST] and is valid after the WR1 bit is self-cleared.

The host transmits the 16-byte search value based on the last ROM value found. These 16 bytes are 0x00 for the first run. The 16 bytes returned contain the new ROM code and are also used to generate the next 16 bytes to transmit. This process is repeated until serial numbers duplicate to find all devices.

13.4.2 Low Power Mode

The 1-Wire module automatically goes into low-power mode whenever it is not communicating with a generic 1-Wire device. The main clock is gated off in low-power mode.

As soon as software writes to any register, the 1-Wire module exits low-power mode.

13.4.3 Clocks

The 1-Wire module takes a main clock as a module input and passes it through a clock divider. (See the block diagram in Figure 13-1.) Software must program the divider factor to generate a 1-MHz clock that is used as an internal time base for the module, as given by Equation 13-1.

$$\text{time\_base} = \text{main\_clock} \div (\text{TIME\_DIVIDER[DVDR]} + 1) \qquad \textit{Eqn. 13-1}$$

For example, if the main clock frequency is 30 MHz, the value to write to the divider register is 29. If the main clock input frequency is not an integer, the programmer must ensure the time base frequency is within the range given by Equation 13-2.

Eqn. 13-2

$$0.98\ \text{MHz} \leq \text{time\_base} \leq 1.02\ \text{MHz}$$

NOTE

A main clock frequency below 10 MHz causes improper function of the module.

13.4.4 Reset

The 1-Wire module supports two levels of reset: hardware and software.

13.4.4.1 Hardware Reset

Whenever a device reset occurs, a hard reset is performed on the 1-Wire module, clearing all values written to all registers.

13.4.4.2 Software Reset

Software initiates a software reset by setting the reset bit RESET[RST]. A software reset clears all data written to the registers.

Note that the reset register (RESET) itself is not cleared during a software reset. Software must clear the RST bit to release the software reset.

Chapter 14
Cortex-A8 Platform

14.1 Overview

A block diagram of the Cortex-A8n Core Platform is shown in Figure 14-1. The platform consists of the ARM Ltd. Cortex-A8n processor which includes a NEON co-processor, an L1 cache, an L2 cache, an ETM, and a CTI. The platform includes the essential sub-blocks: platform control, test logic, and the debug modules (CTM, ETB, and a second CTI).

The Cortex-A8n processor instruction and data read/write AXI master port is connected from the non-CPU side of the Level 2 Cache. The L2 can in turn can access L3 memory via this port.

The core platform supports static debug through the debug logic to SOC. This includes the capability of real time trace via ARM's Coresight ETM, ETB, and CTM modules. The second CTI module allows cross triggering of internal and external trigger sources.

The Cortex-A8n platform has two power domains (LP and GP), which are separated by level shifters. The LP power domain serves as an interface to the rest of the SoC. The GP power domain is completely contained within the platform and allows the Cortex-A8n core and subsystem to run at much higher frequencies than the rest of the SoC.

The boundary of the two power domains is also an asynchronous boundary between the Cortex-A8n platform and the rest of the SoC. Synchronizers in both the GP and LP power domains of the platform allow the Cortex-A8n core and subsystem to run asynchronously from the rest of the SoC.

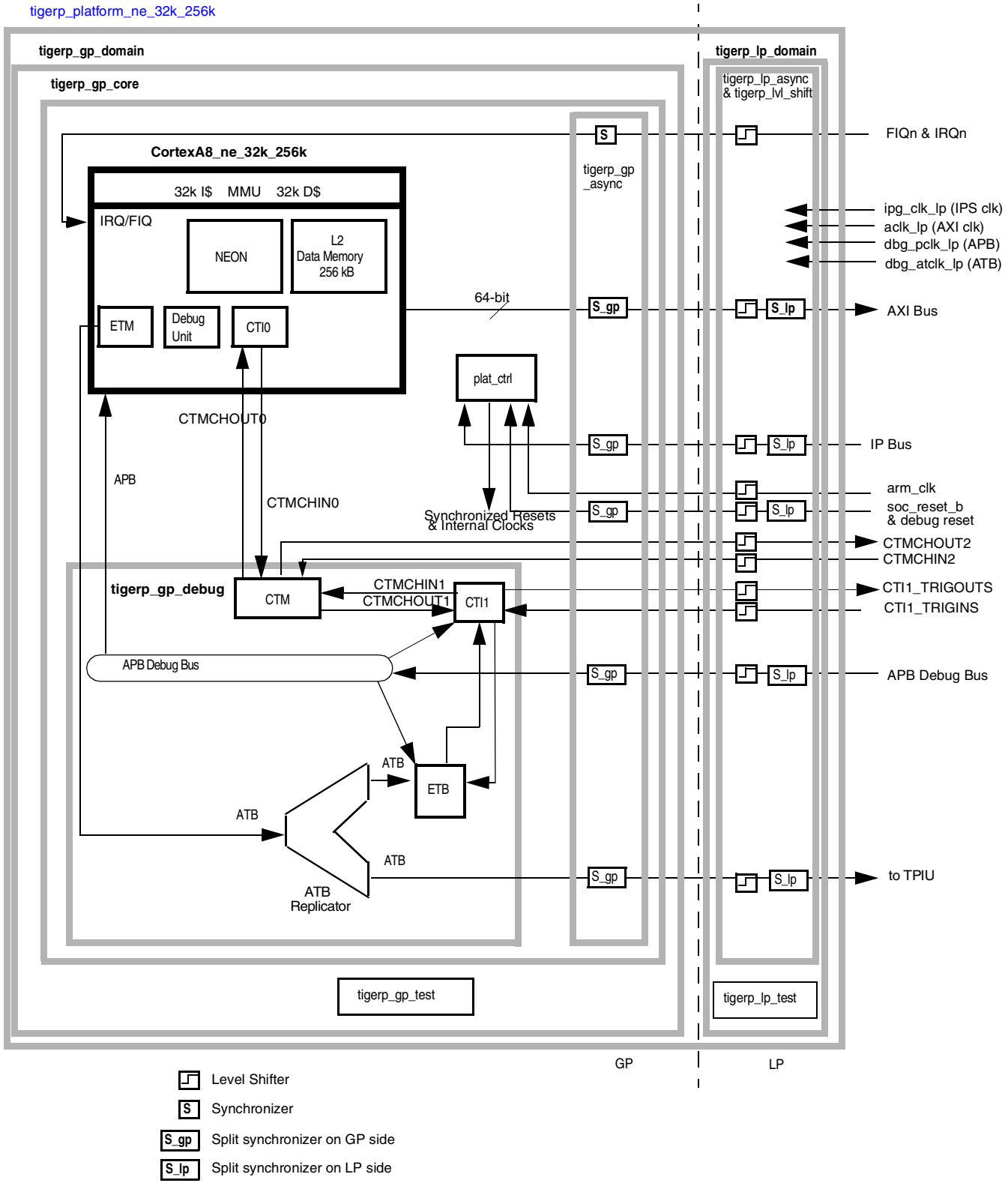


Figure 14-1. Cortex-A8n Core Platform Block Diagram

14.2 Core Platform Sub-blocks

This section discusses the core platform sub-blocks.

14.2.1 Cortex-A8n Processor

The information presented in this section focuses on design aspects of the Cortex-A8n Processor in the AP subsystem. The Cortex-A8n Processor is ARM's first superscalar processor featuring technology for enhanced code density and performance, NEON technology for multimedia and signal processing, and Jazelle RRCT (Runtime Compilation Target) technology for efficient support of ahead-of-time and just-in-time compilation of Java and other bytecode languages.

The Cortex-A8n Processor incorporates an integer core that implements the ARMv7-A architecture instruction set. It supports the ARMv7 and Thumb-2 instruction sets. A NEON module is included to accelerate the performance of multimedia applications. The included Vector Floating Point v3 architecture complies with the IEEE 754 standard. The processor includes an AMBA® 3 AXI high-performance 64-bit SoC interconnect.

14.2.1.1 Features

The following list discusses the main features.

- The ARM Cortex-A8n Processor's sophisticated pipeline architecture is based on dual, symmetric, in-order issue, 13-stage pipelines with advanced dynamic branch prediction achieving 2.0 DMIPS/MHz. The instruction execute unit consists of two symmetric *Arithmetic Logical Unit* (ALU) pipelines and the multiply pipeline.
  - In-order, dual-issue, superscalar microprocessor core
  - 13-stage main integer pipeline
  - 10-stage NEON media pipeline for executing NEON and VFP instruction sets
  - Dedicated L2 cache with programmable wait states
  - Global history based branch prediction
- Works in conjunction with a power optimized load store pipeline to deliver 2.0 DMIPS/MHz for power sensitive applications
- ARMv7 architecture compliant including:
  - Thumb®-2 technology for greater performance, energy efficiency, and code density
  - NEON signal processing extensions to accelerate media codecs such as H.264 and MP3
  - Jazelle RCT Java-acceleration technology to optimize Just In Time (JIT) and Dynamic Adaptive Compilation (DAC), and reduce memory footprint by up to three times
  - TrustZone™ technology for secure transactions and Digital Rights Management (DRM)
- Integrated Level 2 Cache
  - Built using standard compiled LP RAMs
  - Sized at 256 Kb
  - Programmable delay

- Optimized Level 1 Caches
  - Customized design for performance and power optimization
  - Sized at 32 Kb Instruction and 32 Kb Data
  - Combine minimal access latency with hash way determination to maximize performance and minimize power consumption.
- Dynamic Branch Prediction
  - Enabled by branch target and global history buffers
  - Achieves 95% accuracy across industry benchmarks.
  - Replay mechanism minimizes miss-predict penalty
- Memory System
  - Single-cycle load-use penalty for access to the L1 cache
  - Hash array in the L1 cache limits activation of the memories to when they are likely to be needed.
  - Direct interface between the integrated, configurable L2 cache and the NEON media unit for data streaming
  - Banked L2 cache design that enables only one bank at a time
- Memory Management Unit (MMU) and separate instruction and data Translation Look-aside Buffers (TLBs) of 32 entries each
- Embedded Trace Macrocell (ETM) support for non-intrusive debug
- ARMv7 debug with watchpoint and breakpoint registers and a 32-bit Advanced Peripheral Bus (APB) interface to the CoreSight debug system.

14.2.1.2 Instruction Fetch

The instruction fetch unit predicts the instruction stream, fetches instructions from the L1 instruction cache, and places the fetched instructions into a buffer for consumption by the decode pipeline. The instruction fetch unit also includes the L1 instruction cache.

14.2.1.3 Instruction Decode

The instruction decode unit decodes and sequences all ARM and Thumb-2 instructions including the debug control coprocessor, CP14, and the system control coprocessor, CP15 instructions.

The instruction decode unit handles the sequencing of the following:

- Exceptions
- Debug events
- Reset initialization
- *Memory Built-In Self Test* (MBIST) for L1 cache
- Wait-for-interrupt
- Other unusual events
- *Instruction Cycle Timing*

14.2.1.4 Instruction Execute

The instruction execute unit consists of two symmetric Arithmetic Logical Unit (ALU) pipelines and the multiply pipeline. The execute pipelines also perform register write back.

The instruction execute unit performs the following tasks:

- Executes all integer ALU and multiply operations including flag generation
- Generates the virtual addresses for loads and stores and the base write-back value, when required
- Supplies formatted data for stores and also forwards data and flags
- Processes branches and other changes of instruction stream and evaluates instruction condition codes.

14.2.1.5 Load/Store

The load/store unit encompasses the entire L1 data side memory system and the integer load/store pipeline. This includes the following:

- L1 data cache
- Data side TLB
- Integer store buffer
- NEON store buffer
- Integer load data alignment and formatting
- Integer store data alignment and formatting.

The pipeline accepts one load or store per cycle that can be present in either pipeline 0 or pipeline 1. This gives the processor flexibility when scheduling load and store instructions.

14.2.1.6 L2 Cache

The L2 cache unit includes the L2 cache and the Buffer Interface Unit (BIU). It services L1 cache misses from both the instruction fetch unit and the load/store unit.

14.2.1.7 NEON

The NEON unit includes the full 10-stage NEON pipeline that decodes and executes the NEON media instruction set. The NEON unit includes the following:

- NEON instruction queue
- NEON load data queue
- Two pipelines of NEON decode logic
- Three execution pipelines for NEON integer instructions
- Two execution pipelines for NEON floating-point instructions
- One execution pipeline for NEON and VFP load/store instructions
- VFP engine for full execution of the VFPv3 data-processing instruction set.

14.2.1.8 Processor Debug Unit

The processor debug unit assists in debugging software running on the processor. In combination with a software debugger program, the debug unit enables debugging the following:

- Application software
- Operating systems
- Hardware systems based on an ARM processor

The debug unit enables the following:

- Stopping program execution
- Examining and altering processor and coprocessor states
- Examining and altering memory and input/output peripheral states
- Restarting the processor core

14.2.1.9 Embedded Trace Macrocell (ETM)

The ETMv3.3 unit is a nonintrusive trace macrocell that filters and compresses an instruction and data trace for use in system debugging and system profiling. The ETM unit has an external interface outside of the processor called the Advanced Trace Bus (ATB) interface.

The Cortex-A8n Processor ETM provides real time instruction trace for the Cortex-A8n Processor. It is designed to be used with the CoreSight Design Kit. Unlike previous versions of the ARM platforms, this ETM is embedded inside the Processor Core.

Real time tracing is controlled by specifying a set of filtering and triggering resources which include address and data comparators, counters and sequencers. Note that although data trace cannot be enabled, the ETM can still trigger based on data values. Also, the ETM can trace data address values.

The Cortex-A8n Processor ETM contains the following main components:

- Core interface
- Trace generation
- Filtering and triggering resources
- Main FIFO
- AMBA 3 ATB interface
- AMBA 3 APB interface

14.2.1.10 Cross Trigger Interface 0 (CTI0)

This block controls the Trigger Interface (TI). The CTI combines and maps the trigger requests, and broadcasts them to all other interfaces on the ECT as channel events. When the CTI receives a channel event it maps this onto a trigger output. This enables subsystems to cross trigger with each other. The receiving and transmitting of triggers is performed through the TI.

Each CTI has eight trigger inputs, eight trigger outputs, and four sets of 4-channel I/O(s).

14.3 Summary of Remaining Platform Components

This section provides a summary of the platform components that have not been previously discussed.

14.3.1 Platform Controller

The Platform Control contains all the miscellaneous logic required for the platform. The main purpose of the Platform Control module within the Cortex-A8n Core Platform is to implement a set of control and/or status registers and associated logic for the management of certain platform functions such as internally generated clocks, debug enable/status, low-power control, platform version ID, and others.

The Platform Control Module provides these main features:

- Simple IPbus interface with TrustZone type security to all registers.
- Internal clock generation with the ability to programatically set the frequency (divide-by) for each clock independently.
- Ability to force (down-counter) preload for any one or more clocks at anytime independent of preload set.
- Provides a gated clock for the IPbus asynchronous bridge for power savings.
- Provides 16 bits of General Purpose register bits for routing out to Platform boundary.
- Provides 8 bits of Platform Internal Control register bits for external Platform (SoC) use.
- Provides a set of Low Power control and status bits.
- Control bit to enable Debug along with Debug active status bit.
- NEON activity monitor which can be used in determining when to disable NEON thus saving power.

14.3.1.1 Debug Sub-Blocks

The Cortex-A8n Core Platform debug blocks are part of the overall Coresight debug system, which includes the ETB, CTM, CTI1, ATB replicator, and APB address decode. It is expected that a DAP module and one or more additional CTI will be included at the SoC level. This section gives a brief overview of the modules that are implemented within the Cortex-A8n Core Platform platform. For details of the full Coresight debug subsystem, please refer to the SoC debug section.

14.3.1.1.1 Embedded Trace Buffer (ETB)

The ETB provides on-chip storage of trace data using 32-bit RAM. The ETB accepts trace data from the Cortex-A8n ETM via an ATB port (passing through a replicator in between). Providing an on-chip buffer alleviates the pin count, bandwidth, and pad design requirements associated with sending trace data to a debugger directly through package pins in a real-time fashion.

The features are as follows:

- 4kB compiled memory for the trace buffer and optionally can be used as a general purpose memory
- Only 32-bit accesses to the 4kB buffer is supported
- AMBA Peripheral Bus programming interface for configuration and memory access

14.3.1.1.2 AMBA Trace Bus (ATB) Replicator

The ATB Replicator enables two trace sinks (ETB and an off platform port generally connected to a Trace Port Interface Unit—TPIU) to be wired together and receive ATB trace data from the same trace source (ETM). There are no programmable registers. It takes incoming trace data from a single source (ETM) and replicates it as multiple masters.

The CSREPLICATOR is part of the ARM platform. Two output master ports are required for this design—one for the on-platform ETB and one for the off-platform TPIU. Placing the TPIU off platform allows future debug trace sources to connect to the TPIU via a FUNNEL without the need to modify the ARM platform.

14.3.1.1.3 Cross Trigger Interface 1 (CTI1)

This block controls the Trigger Interface (TI). The CTI combines and maps the trigger requests as well as broadcasts them to all other interfaces on the ECT as channel events. When the CTI receives a channel event, it maps this onto a trigger output. This enables subsystems to cross trigger with each other. The receiving and transmitting of triggers is performed through the TI.

Each CTI has eight trigger inputs, eight trigger outputs, and four sets of 4-channel I/O(s).

14.3.1.1.4 Cross Trigger Matrix—CTM

The CTM controls the distribution of channel events. It provides channel interfaces to the CTIs. The CTM can also connect to another CTM via a channel interface. This allows multiple CTMs to be connected.

The Cortex-A8n Core Platform has one CTM inside the platform to handle events between the platform's CTI, the processor's CTI and the rest of the ECT system outside the platform. Placing a CSCTM on platform minimizes the event cycle time between the ETB and the ETM. The handshaking between the CTIs on-platform and the on-platform CTM are not enabled. This requires the Cortex-A8n's CTI, the on-platform CTI1, and the on-platform CTM to all be in the same clock domain. The on-platform CTM connects to the ECT system via an off-platform CTM and the respective channels require synchronizing and handshaking enabled.

14.3.1.1.5 Advanced Peripheral Bus—APB Debug Bus

The APB originates off platform generally from a Debug Access Port—DAP block. This bus allows access to the registers in all of the debug modules that have addressable registers.

14.3.1.2 Asynchronous Wrapper

The Cortex-A8n Core Platform hard macro contains synchronizers for most signals into and out of the platform where synchronization is required. This means the Cortex-A8n Core Platform can function asynchronously from the rest of the SoC.

14.3.2 Configuration

There are several configuration options associated with the Cortex-A8n Processor. These are determined at the platform level and selected as follows:

- The L1 cache size is 32 Kb instruction and 32 Kb data
- There is parity on the L1 cache
- There is no error correction on the L1 cache
- The L2 cache size is 256 Kb
- There is no parity on the L2 cache
- There is no error correction on the L2 cache
- There are 2 L2 tag banks and 4 data banks per tag bank
- The AXI data bus width out of the platform is 64-bit

14.3.3 Endian Modes

The Cortex-A8n Core Platform only supports little-endian mode.

14.3.4 Bus Interfaces

This section discusses the major bus interfaces of the Cortex-A8n Core Platform. The Cortex-A8n Processor has the following bus interfaces:

- AMBA AXI interface
- APB CoreSight interface
- ATB CoreSight interface
- Peripheral Interface (IP Bus)

14.3.4.1 AMBA AXI Interface

The AXI bus interface is the main interface to the system bus. It performs L2 cache fills and non-cacheable accesses for both instructions and data. The AXI interface utilizes a 64-bit wide input and output data buses. It also supports multiple outstanding requests on the AXI bus. The AXI bus is allowed to be asynchronous between the platform and SoC domains. This provides maximum flexibility at the SoC level.

14.3.4.2 APB CoreSight Interface

The APB is an AMBA bus used for debugging. The CoreSight interface is the ARM architecture for multi-processor trace and debug. It defines what debug and trace components are required and how they are connected. The platform Debug APB should be connected to the SoC Debug Access Port (DAP) APB mux. The APB is allowed to be asynchronous between the platform and SoC domains. This provides maximum flexibility at the SoC level.

14.3.4.3 ATB CoreSight interface

The ATB is a trace output bus used for debugging. The CoreSight components are programmed with the *Debug Access Port* (DAP) using the APB programming bus. Trace is output over the ATB trace bus. The ATB is allowed to be asynchronous between the platform and SoC domains. This provides maximum flexibility at the SoC level.

14.3.4.4 Peripheral Interface (IP Bus)

The IP Bus interface to the platform connects the on-chip registers to the memory map and internal buses at the SoC level. Location (addresses of the registers) is determined at the SoC level. The platform acts as a slave to an SoC-based IP Bus controller.

14.4 Memory Map and Register Definition

This section provides a register memory map and register summary.

14.4.1 Register Memory Map

The memory map for Cortex-A8n Core Platform specific registers is shown in Table 14-1. This does not include the co-processor registers contained in the ARM Cortex-A8n Processor or the ARM Coresight Debug. For documentation of registers contained in those modules, please refer to the appropriate technical reference manual.

This register block address space uses 5 address lines which are fully decoded. The space includes 9 registers and 23 unimplemented locations. The unimplemented locations return an error if they are accessed. Depending on how the SoC decodes the register block, the entire 32-word register block could alias in the larger memory space.

Table 14-1. Block Memory Map

Address	**Register**	Access	Reset Value	Section/Page
General Registers				
0xBASE_0000 (PVID)	Platform Version ID	R/	0x????_????	14.4.3.1/14-13
0xBASE_0004 (GPC)	General Purpose Control	R/W	0x0000_FF00	14.4.3.2/14-14
0xBASE_0008 (PIC)	Platform Internal Control	R/W	0x0000_00F0	14.4.3.3/14-16
0xBASE_000C (LPC)	Low Power Control	R/W	0x0000_0000	14.4.3.4/14-16
0xBASE_0010 (LPC)	NEON Low Power Control	R/W	0x0000_0000	14.4.3.5/14-18
0xBASE_0014 (ICGC)	Internal Clock Generation Control	R/W	0x0000_7777	14.4.3.6/14-18

Table 14-1. Block Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE_0018 (AMC)	ARM Memory Configuration	R/W	0x0000_0003	14.4.3.7/14-21
0xBASE_0020 (NMC)	NEON Monitor Control	R/W	0x000F_F000	14.4.3.8/14-22
0xBASE_0024 (NMS)	NEON Monitor Status	R/W	0x0000_0000	14.4.3.9/14-23

14.4.2 Register Summary

The conventions in Figure 14-2 and Table 14-2 serve as a key for the register summary and individual register diagrams.

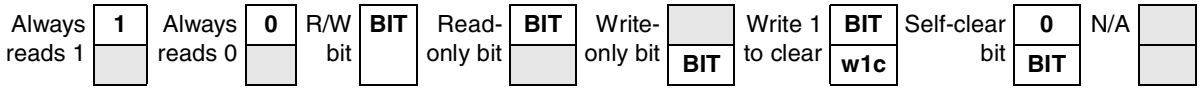


Figure 14-2. Key to Register Fields

Table 14-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
?	Unknown at publication time and subject to change.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

14.4.3 Register Descriptions

Table 14-3 shows the core platform register summary.

Table 14-3. Core Platform Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE_0000 (PVID)	R	SPEC[31:24]								IMPL[23:16]							
	W																
	R	MINOR[15:8]								ECO[7:0]							
	W																
0xBASE_0004 (GPC)	R	DBG ACTIVE	0	0	0	0	0	0	0	0	0	0	0	0	NO CLK STP	AT RDY	DBG EN
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
0xBASE_0008 (PIC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	PIC[7:0]							
	W																
0xBASE_000C (LPC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DBG DSM	DSM
	W																
0xBASE_0010 (NLPC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NEON RST
	W																
0xBASE_0014 (ICGC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	DP_CLK_DIVR [2:0]			0	ACLK_DIVR[2:0]			0	IPG_CLK_DIVR [2:0]		
	W					DT_PRLD				ACLK_PRLD				IPG_PRLD			

Table 14-3. Core Platform Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE_0018 (AMC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	ALP EN	ALP[2:0]		
	W																
0xBASE_0020 (NMC)	R	IE	NME	0	0	0	0	0	0	0	0	0	0	PL[19:16]			
	W																
	R	PL[15:12]				0	0	0	0	0	0	0	0	0	0	0	0
	W																
0xBASE_0024 (NMS)	R	NI	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W	w1c															
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																

NOTE

All registers are zero wait-state and all accesses must use a 32-bit word size. Any other transaction sizes produce a zero wait-state error response.

14.4.3.1 Platform Version ID Register

The platform version ID register (PVID) contains a 32-bit version ID which can be used to link the silicon to a specific version of the platform database. The version ID should match our release label applied to the platform database.

The release label uses the following convention:

TIGERP.C65GP.XX.XX.XX.XX.

This register can be accessed by a 32-bit secure/non-secure, user/supervisor, read transaction. Writes return an error.

0xBASE_0000 (PVID)

Access: User/Supv, Secure/non-Secure Read

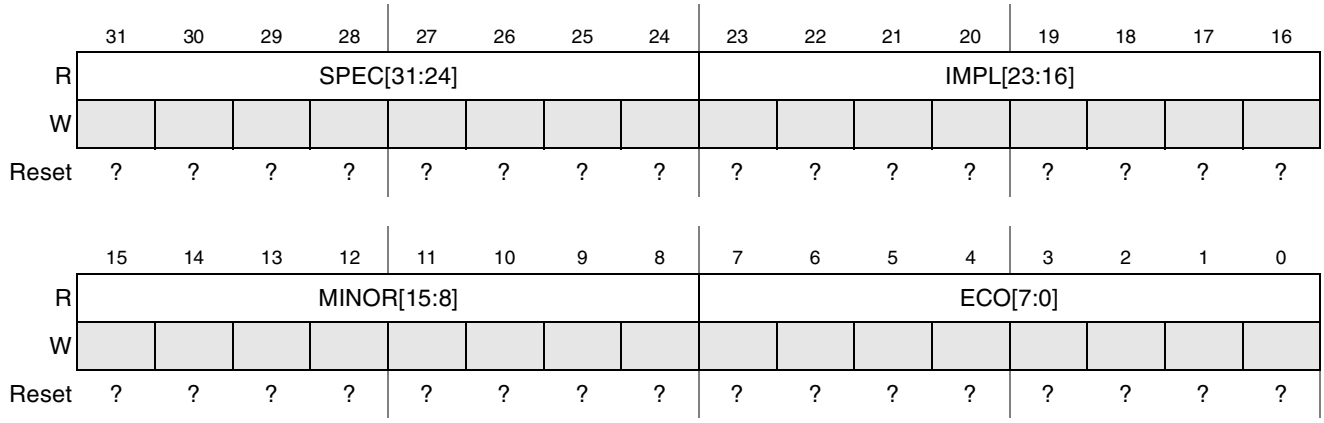


Figure 14-3. Platform Version ID Register

Table 14-4. PVID Field Descriptions

Field	Description
31–24	SPEC[31:24] = Major architectural or significant spec changes The reset value is unknown at publication of this document and is also subject to change.
23–16	IMPL[23:16] = Implementation changes The reset value is unknown at publication of this document and is also subject to change.
15–8	MINOR[15:8] = Minor changes (bug fixes, I/O changes) The reset value is unknown at publication of this document and is also subject to change.
7–0	ECO[7:0] = ECO changes The reset value is unknown at publication of this document and is also subject to change.

14.4.3.2 General Purpose Control Register

This register contains 16 bits of general purpose control which drive core platform outputs that can be used within the SoC for general purpose control. This register also contains the DBG_ACTIVE status bit and DBG_EN control bit for entering Debug mode via a software access to this register.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_0004 (GPC)

Access: Secure, Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DBG ACTIVE	0	0	0	0	0	0	0	0	0	0	0	0	NO CLK STP	AT RDY	DBG EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Figure 14-4. General Purpose Control Register

Table 14-5. GPC Field Descriptions

Field	Description
31	DBGACTIVE - This bit indicates the status of debug. This allows the user to determine if debug has been enabled either from off platform, via the external DAP_SYS JTAG interface (dbgen_in platform input), or via software accesses to the DBGEN bit in this register. If 0, debug is not enabled. Debug clocks are off and debug registers are inaccessible. If 1, debug is enabled. Debug clocks are on and the debug system is ready to be used.
30–19	Reserved
18	NOCLKSTP - This bit is used to control clock-gating within the CORTEX-A8n. Note that this is distinctly different than the higher-level (platform-level) clock-gating/low-power control afforded by the other Plat_Ctrl low-power functionality (i.e. **dsm_request** along with the **DSM** and **DBG_DSM** control bits within the **Low Power Control** register). If 1 (reset state), then all normal clock-gating activity (WFI, NEON, etc.) is overridden and no clock gating occurs within the CORTEX-A8n. **NOCLKSTP**, however, has no affect on the higher-level platform clock-gating, and the low-power functions discussed in the **LPC** register work as architected. if 0, then clock-gating within the CORTEX-A8n is enabled. This is the recommended setting in order to reduce run mode power consumed by the clocking network within the Integer core as well as the Neon co-processor.
17	ATRDY - If 1, this bit disables the platform boundary ATB interface. In most SoCs, this is connected to a TPIU. This prevents traces to the ETB from stalling due to the ASYNCATB FIFO overflowing. If 0, this bit allows the platform boundary ATB interface to be active. In this case, traces to the ETB could stall due to ASYNCATB FIFO becoming full. This may cause the ETM data loss.
16	DBGEN - Debug enable. This allows the user to manually activate clocks within the debug system. This register bit directly controls the platform's dbgen_out output signal which connects to the DAP_SYS to enable all debug clocks. Once enabled, the clocks cannot be disabled except by asserting the disable_trace input of the DAP_SYS.
15-0	Reserved

14.4.3.3 Platform Internal Control Register

This register contains eight bits that drive internal signals which can be used within the platform for general purpose control. Currently, none of the PIC bits are being utilized.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_0008 (PIC)

Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PIC[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Figure 14-5. Platform Internal Control Register

14.4.3.4 Low-Power Control Register

This register is used to control entry to low-power mode and can only be accessed by 32-bit secure supervisor transactions.

There exists a platform output, **dsm_request**, that when asserted, requests the SoC to turn off clocks to the platform, and optionally deactivate one or more power supplies. DSM can only be entered after the CPU has executed a WFI instruction, completed all outstanding bus transactions, and all activity at the L2 level has completed. Assertion of either **irq_b**, **fiq_b**, or **dbg_edbgrq** platform inputs cause the CPU to complete the WFI instruction and result in the negation of **dsm_request**.

The DSM bit configures the platform to either: block deep sleep mode entry (default), or allow DSM entry when requested.

The DBG_DSM bit configures the platform to block deep sleep mode entry when debug mode is enabled (dbgen_in platform input is asserted high). This allows the user to ensure deep sleep mode is not entered when debug is enabled.

The NEON_RST bit is used for placing the Cortex-A8n NEON processor into, or releasing it from, reset.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_000C (LPC)

Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DBG DSM	DSM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-6. Low-Power Control Register

Table 14-6. LPC Field Descriptions

Field	Description
31–2	Reserved
1	DBGDSM - Debug Deep Sleep Mode Enable If 1, DSM can be entered (dsm_request will not be blocked) regardless if debug is enabled or not. If 0, DSM can NOT be entered (dsm_request will be blocked) if debug is enabled
0	DSM - Deep Sleep Mode Enable is used to gate the platform's dsm_request signal, therefore preventing the platform from issuing a deep sleep mode request. If 1, DSM can be entered (dsm_request will not be blocked). If 0, DSM can NOT be entered (dsm_request will be blocked).

14.4.3.5 NEON Low-Power Control Register

This register is used to place the Cortex-A8n NEON processor into, or release it from, reset. Refer to Table 14-7 for a description of this control bit.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_0010 (NLPC) — Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0		NEON RST
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-7. Low Power Control Register

Table 14-7. NLPC Field Descriptions

Field	Description
31-1	Reserved
0	NEONRST - Hold the Cortex-A8n NEON in it's reset state in preparation for a low-power mode. • If 1, the ARESETNEONn input to the Cortex-A8n is driven low - placing NEON into reset. • If 0, the ARESETNEONn input to the Cortex-A8n is driven high - releasing NEON from reset.

14.4.3.6 Internal Clock Generation Control Register

This register is used to control the clock generation circuitry for all derived clocks used within the ARM platform (**ipg_clk**, **aclk**, **dbg_atclk**, and **dbg_pclk**). The **dbg_pclk** clock is generated as a straight divide-by-2 of **dbg_atclk**.

NOTE

At reset, all generated clocks are edge aligned and are generated at 8:1. The **dbg_pclk** signal requires 1 **dbg_atclk** clock cycle before it begins clocking as a divided **dbg_atclk**.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_0014 (ICGC)

Access: Secure, Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	DT_CLK_DIVR[2:0]			0	ACLK_DIVR[2:0]			0	IPG_CLK_DIVR[2:0]		
W					DT_PRLD				ACLK_PRLD				IPG_PRLD			
Reset	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1

Figure 14-8. Internal Clock Generation Control Register

Table 14-8. ICGC Field Descriptions

Field	Description
31-12	Reserved
11	DT_PRLD - Debug AMBA Trace Bus Clock Down Counter Preload Reads always return 0 A write of 0 has no effect. If preload is 0 during a write of a new divide ratio, the down counteris not loaded with the new value until it reaches 0 (i.e. rolls over). A write of 1 forces the DBG_ATCLK down counter to preload on the next arm_clk.
10-8	DT_CLK_DIVR[2:0] - Debug AMBA Trace Bus Clock Divide Ratio These bits control the clock divide ratio for the debug AMBA trace bus (ATB) and GP async interface clocks. The ATB bus is used to transfer trace messages from the ETM to the ETB or TPIU. When ETM trace data is being transferred over the ATB to the embedded trace buffer (ETB), the ATB is capable of running up to half the CPU frequency. However, when ETM trace data is being transferred to the TPIU to be sent out the trace port, the ATB should be slowed to some appropriate SoC frequency. DT_CLK_DIVR[2:0] = 0b000 results in a 1:1 clock ratio DT_CLK_DIVR[2:0] = 0b001 results in a 2:1 clock ratio DT_CLK_DIVR[2:0] = 0b010 results in a 3:1 clock ratio DT_CLK_DIVR[2:0] = 0b011 results in a 4:1 clock ratio DT_CLK_DIVR[2:0] = 0b100 results in a 5:1 clock ratio DT_CLK_DIVR[2:0] = 0b101 results in a 6:1 clock ratio DT_CLK_DIVR[2:0] = 0b110 results in a 7:1 clock ratio DT_CLK_DIVR[2:0] = 0b111 results in a 8:1 clock ratio
7	ACLK_PRLD - AXI Master Port Clock Down Counter Preload Reads always return 0 A write of 0 has no effect. If preload is 0 during a write of a new divide ratio, the down counter is not loaded with the new value until it reaches 0 (i.e. rolls over). A write of 1 forces the ACLK down counter to preload on the next arm_clk.

Table 14-8. ICGC Field Descriptions (continued)

Field	Description
6-4	ACLK_DIVR[2:0] - AXI Master Port Clock Divide Ratio Controls the clock divide ratio for the Tiger CPU AXI bus and GP AXI async interface clocks and clock enable. The ratio of clock cycles generated for arm_clk to aclk is: ACLK[2:0] +1 : 1 ACLK_DIVR[2:0] = 0b000 results in a 1:1 clock ratio ACLK_DIVR[2:0] = 0b001 results in a 2:1 clock ratio ACLK_DIVR[2:0] = 0b010 results in a 3:1 clock ratio ACLK_DIVR[2:0] = 0b011 results in a 4:1 clock ratio ACLK_DIVR[2:0] = 0b100 results in a 5:1 clock ratio ACLK_DIVR[2:0] = 0b101 results in a 6:1 clock ratio ACLK_DIVR[2:0] = 0b110 results in a 7:1 clock ratio ACLK_DIVR[2:0] = 0b111 results in a 8:1 clock ratio
3	IPG_PRLD - Platform IP-Bus Port Clock Down Counter Preload Reads always return 0 A write of 0 has no effect. If preload is 0 during a write of a new divide ratio, the down counter is not loaded with the new value until it reaches 0 (i.e. rolls over). A write of 1 forces the IPG_CLK down counter to preload on the next arm_clk.
2-0	IPG_CLK_DIVR[2:0] - Platform IP-Bus Port Clock Divide Ratio Controls the clock divide ratio for the Platform Controller and GP async IP-Bus clock and clock enable. The ratio of clock cycles generated for arm_clk to ipg_clk is: IPG_CLK[2:0] +1 : 1 IPG_CLK_DIVR[2:0] = 0b000 results in a 1:1 clock ratio IPG_CLK_DIVR[2:0] = 0b001 results in a 2:1 clock ratio IPG_CLK_DIVR[2:0] = 0b010 results in a 3:1 clock ratio IPG_CLK_DIVR[2:0] = 0b011 results in a 4:1 clock ratio IPG_CLK_DIVR[2:0] = 0b100 results in a 5:1 clock ratio IPG_CLK_DIVR[2:0] = 0b101 results in a 6:1 clock ratio IPG_CLK_DIVR[2:0] = 0b110 results in a 7:1 clock ratio IPG_CLK_DIVR[2:0] = 0b111 results in a 8:1 clock ratio

14.4.3.7 ARM Memory Configuration Register

This register is used to configure the ALP inputs to the ARM platform memories. These bits are used to set the leakage configuration bits on the memories.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE_0018 (AMC)

Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	ALPEN	ALP[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 14-9. ARM Memory Configuration Register

Table 14-9. AMC Field Descriptions

Field	Description
31-4	Reserved
3	ALPEN - ALP Enable is used to activate the ALP bits in this register to override the default memory leakage configuration setting. If 1, the ALP bits of the memory are over-written with the ALP bits of this register. If 0, the ALP bits of the memory are driven to the default (reset) value of 3'b011.
2-0	ALP[2:0] - Memory leakage configuration bits

14.4.3.8 NEON Monitor Control Register

This register is used to control the NEON monitor.

This register can only be accessed by 32-bit secure/non-secure supervisor transactions.

0xBASE_0020 (NMC)

Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IE	NME	0	0	0	0	0	0	0	0	0	0	PL[19-16]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PL[15-12]				0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1

Figure 14-10. NEON Monitor Control Register

Table 14-10. AMC Field Descriptions

Field	Description
31	IE - Interrupt Enable If 1, the monitor interrupt output is enabled and plat_ctrl_nm_irq_b is asserted when the monitor counter expires. If 0, the monitor interrupt output is disabled.
30	NME - NEON Monitor Enable
29-20	Reserved
19-12	PL[19:12] - Preload value for the upper 8 bits of the 16 bit NEON activity counter.
11-0	Reserved

14.4.3.9 NEON Monitor Status Register

This register is used to read the status of the NEON monitor.

This register can only be accessed by 32-bit secure/non-secure supervisor transactions.

0xBASE_0024 (NMS)

Access: Secure, Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NI	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 14-11. NEON Monitor Status Register

Table 14-11. AMC Field Descriptions

Field	Description
31	NI - NEON Idle Status • 1 - Indicates that the NEON activity counter has expired. plat_ctrl_nm_irq_b asserts if the IE bit is set. • 0 - Indicates that the NEON activity counter has not expired.
30-0	Reserved

14.5 Platform Clocks

The clocking strategy of the Cortex-A8n Core Platform can be summarized as follows:

- The Cortex-A8n Core Platform receives one functional clock from the CCM, **arm_clk**, and several SoC clocks for boundary synchronization purposes (**arm_clk** can be completely asynchronous from any other SoC clocks).
- The Cortex-A8n Core Platform, in conjunction with an external Clock Control Module (CCM), supports dynamic clock frequency scaling.
- Divided clocks based on **arm_clk** are generated within the platform for clocking buses which cannot operate at the high **arm_clk** frequencies. These divided clock ratios can be configured by on-platform registers.
  - Derived Clocks:
    - ipg_clk: This clock is used for IP bus logic.
    - aclk: This clock is used for AXI bus logic.
    - dbg_atclk: This clock is used for the Debug AMBA Trace logic.
    - dbg_pclk: This clock is used for the Debug synchronizer logic.
- Two-level clock gating is employed.
  - Module level clock gating is used when possible. A specific module's clock is gated off when the module is not in use.
  - Register level clock gating is used throughout the platform via power design tools.
- **arm_clk** may be turned off for various low-power use cases by an external clock control module that monitors the **DSM_request** output.

14.6 Platform Power Management

The Cortex-A8n Core Platform contains low voltage logic elements, an asynchronous and level shifted interface, state retention latches, low leakage power switches, and floating node isolation circuits. These elements each serves a specific purpose in the power management scheme implemented in the Cortex-A8n Core Platform. The power management capabilities of the Cortex-A8n Core Platform are summarized in this section.

14.6.1 Voltage and Frequency Scaling

The Cortex-A8n Core Platform contains structures necessary for the independent voltage scaling of the GP supply. Voltage and frequency scaling control systems are implemented at the chip level with the Cortex-A8n Core Platform acting as an object of the control. Asynchronous Interface Logic and Level-shifting between SOC-logic and ARM-logic have been implemented in the Cortex-A8n Core Platform to support the voltage and frequency scaling capabilities of the integrated system.

14.6.1.1 Asynchronous Interface Logic

Voltage and frequency scaling is most effective when there is a continuum of available frequencies at which the Cortex-A8n Processor may be clocked. Asynchronous Interface Logic removes the clock ratio dependencies between the Cortex-A8n Processor and the system in which it operates.

14.6.1.2 Level Shifting between SOC-logic and ARM-logic

Independent voltage scaling of the Cortex-A8n Processor requires a distinct power supply for the GP logic. Level shifters exist at the interface of the SOC logic and the ARM logic to allow independent voltage control of the SOC power supply and the ARM power supply.

14.6.2 Power Gating in the Cortex-A8n Core Platform

To reduce standby leakage power consumption, the Cortex-A8n Core Platform contains internal power supplies that may be completely power gated during wait for interrupt modes. Entry and exit sequences of power gating modes are completely controlled by programmed operation of the General Power Controller, so this section only describes what is possible in terms of how the Cortex-A8n Core Platform may be powered down during wait for interrupt modes.

14.6.2.1 Isolation Circuitry at the Cortex-A8n Core Platform Interface

To power down digital logic correctly, a special interface must exist that protects the system from floating signals. Such isolation circuitry has been carefully designed in order to allow glitch-free power down modes of the Cortex-A8n Core Platform.

14.6.2.2 Power Gating the Memory Peripheries

Once the STANDBYWFI mode has been invoked, all of the Cortex-A8n Core Platform memory peripheries may be power gated by the assertion of both the l1_pwrdwn and the l2_pwrdwn signals. This significantly reduces the amount of leakage from the GP supply.

14.6.2.3 Retaining the State of the Cortex-A8n Core Platform Registers

The Cortex-A8n Core Platform has been implemented using State Retention Power Gating (SRPG) register elements. In order to safely enter a power gated mode that includes power gating of the Cortex-A8n Core Platform high performance logic gates, a sequenced assertion of the srpg_pgrst and srpg_pg{0-17} signals must occur. Similarly, a sequenced deassertion is necessary when emerging from power gated states. The required sequences are automatically provided to the Cortex-A8n Core Platform by the General Power Controller.

14.6.2.4 Power Gating the Cortex-A8n Core Platform High Performance Logic Gates

Once the STANDBYWFI mode has been invoked, the Cortex-A8n Core Platform high performance logic gates may be power gated by the assertion of the cpu_powerdown signal and the negation of the vdd_short_b signal. This turns off a portion of the leakage from the GP supply.

14.6.2.5 Power Gating the L2 Bit Arrays

If the L2 cache has been flushed, the L2 bit array may be power gated during the subsequent STANDBYWFI mode. Once the STANDBYWFI mode has been invoked, the L2 memory bit arrays may be power gated by the assertion of the l2bits_powerdown signal. This turns off a portion of the leakage from the LP supply.

14.6.2.6 Controlling Power Gating using the General Power Controller (GPC)

Specific details concerning the power gating entry and exit sequence may be obtained from the block guide of the General Power Controller (GPC). This high level summary is intended to provide some overview.

Code must be formed that takes advantage of low workloads by putting the Cortex-A8n Core Platform into a STANDBYWFI state whenever possible. Prior to entering the STANDBYWFI state, the code should assess several things about the workload and the battery conservation requirements:

- What is the maximum acceptable interrupt service latency?
- What is the expected time to be spent in STANDBYWFI?

Once these time requirements are understood, then Table 14-12 can be used to determine the appropriate STANDBYWFI mode to invoke. The GPC registers can be programmed with the appropriate values to invoke the appropriate STANDBYWFI mode. After GPC programming is complete, the code should execute the STANDBYWFI instruction.

The GPC enforces the appropriate sequence for entering and exiting the programmed power gated modes.

14.6.2.7 Power Gating the NEON Block

The NEON co-processor in the Cortex-A8n Processor can be put into a state retention mode and powered down to reduce current consumption when it is not needed. The sequence is shown in the flow diagram of Figure 14-12. Each box in the diagram is numbered to correspond to the following more thorough description:

1. User Selects NEON to be Powered Down

   The NEON Coprocessor is enabled out of reset and remains powered up unless the user decides to power it down. This method power downs NEON after a user programmable time of NEON inactivity. The current state of the NEON is held in SRPG flops. To select this operation:

   a) The user code programs the amount of time that the NEON needs to be inactive before power down in the NEON Monitor Control Register. Effectively, the user programs the upper 8 bits of a 20-bit counter. This counter runs at the arm_clk rate and so the user can program a timeout from 4K to 1M arm_clk rising edges.

   b) In the same register, the user code enables the NEON Monitor and enables the NEON Monitor Interrupt.

2. Pre-Set Timer Count

   The timer is loaded (automatically by hardware) with the count value chosen by the user.

3. Decision: NEON Instruction Encountered?

If a NEON instruction is encountered the sequence is aborted until the NEON is no longer in use. As long as NEON Instructions are executing, the NEON status bit will remain in the active state and the counter gets forced with the pre-set count.

4. Decrement Timer

   When no NEON instructions are encountered, the timer counts towards the timeout.

5. Decision: Timed Out?

   If the user selected time has lapsed without any NEON activity, then the hardware generates an interrupt.

6. Exception Sequence: Power Down NEON

   The interrupt service routine disables the NEON module with the following sequence:

   a) Software must disable access to the NEON unit using the Coprocessor Access Control Register, see *c1, Coprocessor Access Control Register* on page 3-64 of the Cortex-A8n TRM. All outstanding NEON instructions retire and all subsequent NEON instruction cause an Undefined instruction exception.

```
MRC p15, 0, <Rd>, c1, c0, 2; Read Coprocessor Access Control Register
BIC <Rd>, <Rd>, #0xF00000; Disable access to CP10 and CP11
MCR p15, 0, <Rd>, c1, c0, 2; Write Coprocessor Access Control Register
```

   b) Activate the NEON output clamps.

   c) Place NEON into state retention.

   d) Prepare the NEON memory for power down keeping the state of it’s bits.

   e) Separate the NEON power supplies so that the ones that retain state can continue to have power.

   f) Remove power from the NEON power domain.

7. Decision: NEON Instruction Attempted

   The part continues running regular CPU instructions with the NEON in low power state as long as no NEON instructions are attempted. When a NEON instruction is attempted, an unimplemented instruction exception occurs.

8. Exception Sequence: Power Up NEON

   An exception is taken when a NEON instruction is attempted with the NEON is disabled. The exception routine determines that this is indeed what occurred, enables the NEON module, and waits for the NEON to power up. Returning from the exception causes the NEON instruction to be restarted and run successfully. This is a more detailed description of the sequence:

   a) Power is turned on to the NEON power domain.

   b) Reconnect the NEON power supplies together.

   c) Restore the NEON memory power maintaining the state of it’s saved bits.

   d) Reload the flops with the SRPG saved contents.

   e) Release the NEON output clamps.

   f) Software must enable access to the NEON unit using the Coprocessor Access Control Register, see *c1, Coprocessor Access Control Register* on page 3-64 of the Cortex-A8n TRM.

```
MRC p15, 0, <Rd>, c1, c0, 2; Read Coprocessor Access Control Register
ORR <Rd>, <Rd>, #0xF00000; Enable access to CP10 and CP11
MCR p15, 0, <Rd>, c1, c0, 2; Write Coprocessor Access Control Register
```

g) Return from the exception causing the NEON instruction to be reloaded and executed.

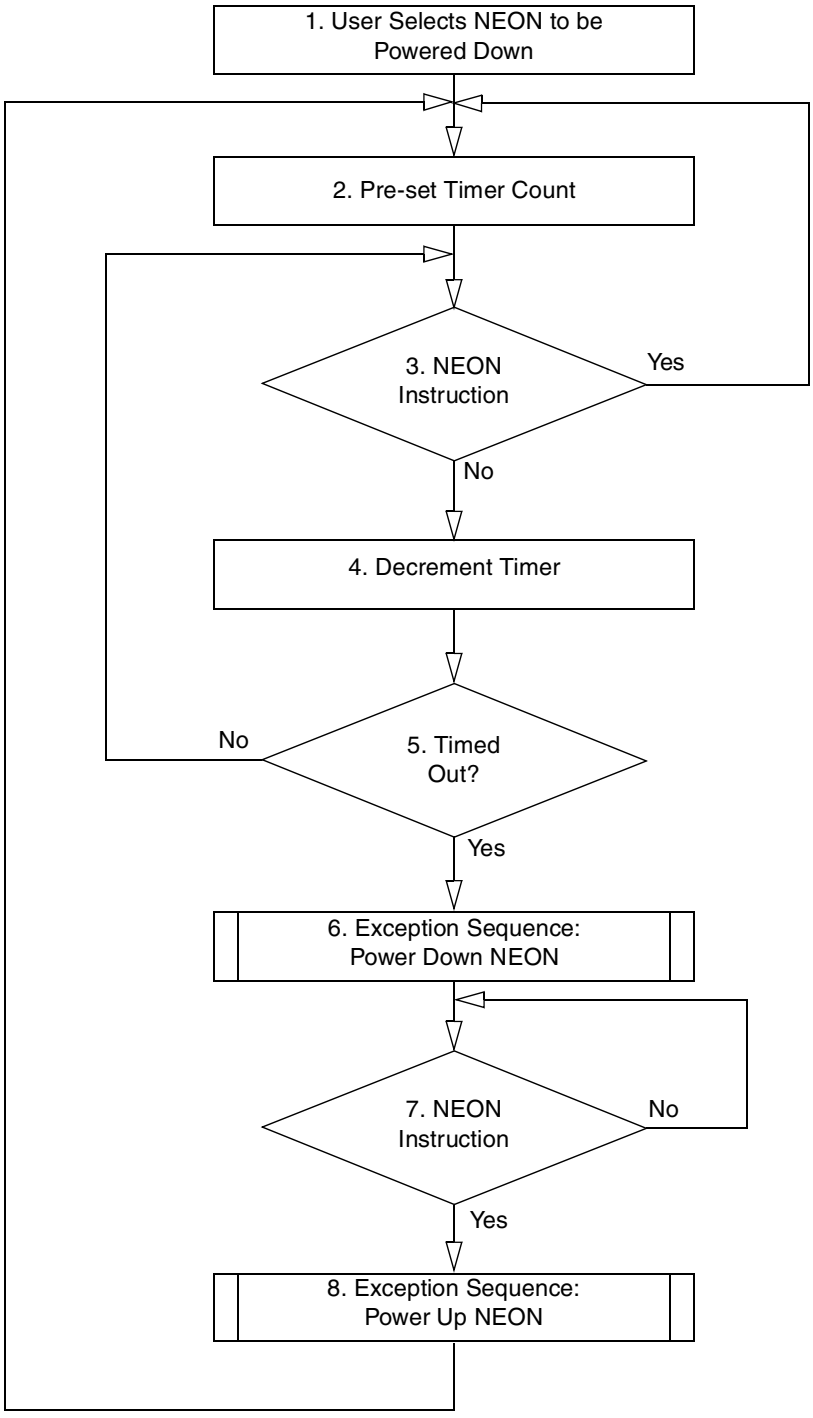


Figure 14-12. Flow Diagram of NEON Power Down Sequence

14.6.3 Modes of Operation

There are several basic low-power modes of the Cortex-A8n Core Platform, as presented in Table 14-12.

Table 14-12. Modes of Operation of the Cortex-A8n Core Platform

MODE	Description
RUN	ARM_CLK running, code executing
CLOCKED_WAIT	ARM_CLK running, STANDBYWFI=TRUE
UNCLOCKED_WAIT	ARM_CLK off, STANDBYWFI=TRUE
STOP1	UNCLOCKED_WAIT plus L1 and L2 peripheries power gated
STOP2	UNCLOCKED_WAIT plus L1 and L2 peripheries power gated, ARM_HiP_logic power gated
STOP3	UNCLOCKED_WAIT plus L1 and L2 peripheries power gated, ARM_HiP_logic power gated, L2 cache flushed, and L2 bit arrays power gated

There are several combinations of reduced power in which one of more portions of the platform are in a power saving mode. This is shown in Table 13.

Table 13. Platform Power Modes

Core and Platform	NEON	L1	L2
Powered Up	Powered Up	Powered Up	Powered Up
Powered Up	Reset (ARM Mode)	Powered Up	Powered Up
State Retention	State Retention	State Retention	State Retention
State Retention	State Retention	Powered Down	State Retention
State Retention	State Retention	State Retention	Powered Down
State Retention	State Retention	Powered Down	Powered Down

Chapter 15
Platform Debug

15.1 Introduction

This chapter provides an overview of the ARM platform debug. It covers the modules inside the tigerp_gp_debug block and the modules that support the platform debug within the SOC, but external to the ARM platform.

Debug for the ARM platform uses ARM's DK11 CoreSight Debug Kit. The following CoreSight modules are used: DAP, ETM, CSREPLICATOR, CSTPIU, CSETB, CSCTI, and the CSCTM. This introduction provides a brief description of each module. For more in-depth information, please refer to the CoreSight_DK_TRM and other relative documentation from ARM.

15.1.1 Overview

ARM's CoreSight Design Kit (CSDK) provides a single solution for multi core and bus trace. The CSDK provides the following capabilities for system-wide trace:

- Debug and trace visibility of the entire system
- Cross-triggering support between SOC subsystems
- Multisource trace in a single stream
- Higher data compression than previous solutions
- Standard programmer's models for standard tools solutions

The CoreSight Design Kit comprises the following main components: control and access, sources, links, and sinks.

- Control and access components configure, access, and control the generation of trace. They neither generate nor process trace data. Examples include the DAP (debug access port) and the ECT (embedded cross trigger interface).
- Source components generate trace data. Example source components are the ETM (embedded trace macrocell) and the future XTM (AXI trace macrocell).
- Link components provide connection, triggering and flow of trace data. Link examples include the ATB replicator and the CoreSight Trace funnel.
- Sink components are the end point for trace data on an SOC. Example sinks are the TPIU (trace port interface unit) and the ETB (embedded trace buffer).

Figure 15-1. shows the ARM Platform debug block diagram. Further detail of the modules shown in the diagram are given in the next sections.

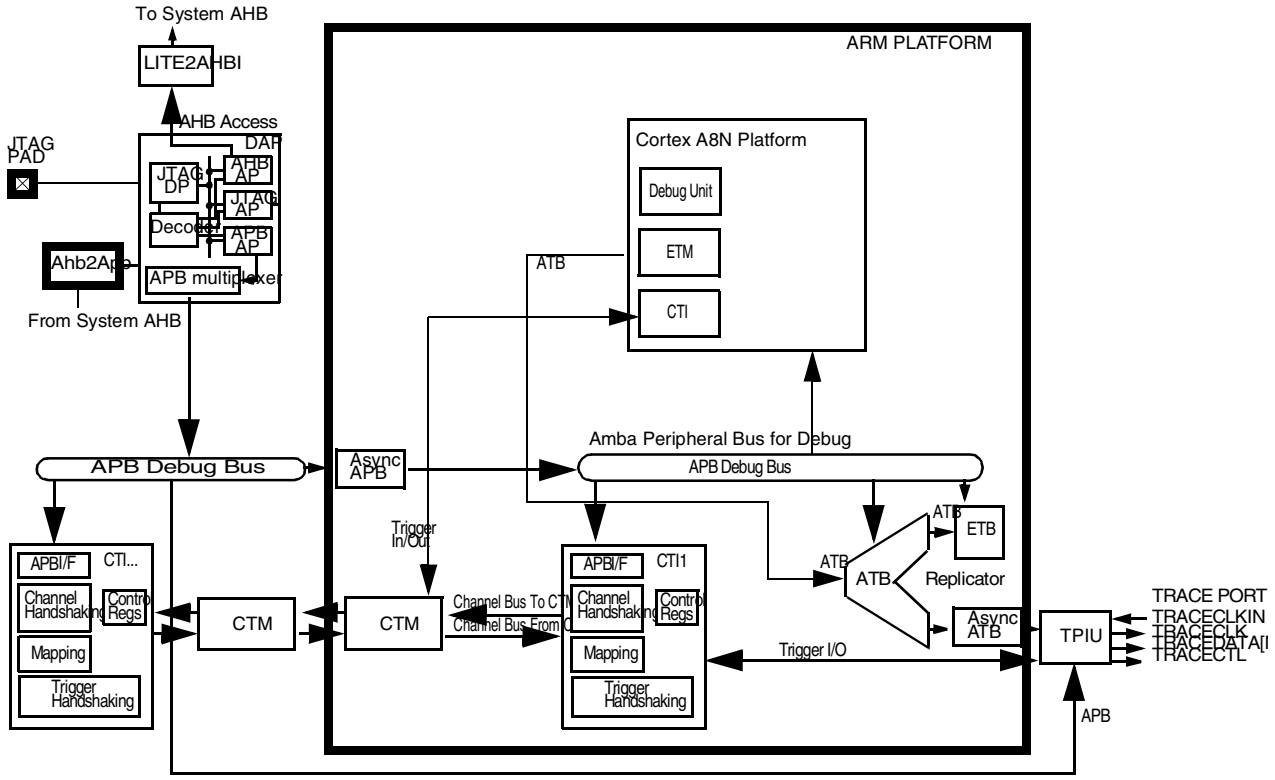


Figure 15-1. ARM Platform Debug Block Diagram

15.1.2 ARM Debug Modules

This section provides an overview of the Debug modules used in the ARM platform and also the modules outside the platform that are included in the CoreSight debug system.

The following modules are included in the ARM platform debug block: CSETB, CSCTI, CSCTM, CSREPLICATOR, and APB address decode logic.

The following modules are part of the ARM platform, outside the debug block: ETMv3.3 (embedded in the Cortex), CTI (also embedded in the Cortex), the Processor Debug Unit, and two asynchronous bridges: the asyncapb and the asyncatb.

Also included in this block guide is an overview of the DAP and TPIU that reside outside the ARM platform, yet are key components to the debug system.

15.1.2.1 Processor Debug Unit

The processor debug unit assists in debugging software running on the processor. In combination with a software debugger program, the debug unit enables debugging the following:

- Application software
- Operating systems
- Hardware systems based on an ARM processor

The debug unit enables the following:

- Stopping program execution
- Examining and altering processor and coprocessor states
- Examining and altering memory and input/output peripheral states
- Restarting the processor core

There are three ways to debug software running on the processor, as follows:

- Halt debug-mode debugging
- Monitor debug-mode debugging
- Trace debugging (ETM) (covered in a different section)

15.1.2.1.1 Halting Debug Mode Debugging

Halting debug-mode debugging is invasive. The processor halts when a debug event, such as a break point, occurs. An external debugger can examine and modify the processor state via the APB while the processor is halted.

15.1.2.1.2 Monitor Debug Mode Debugging

When a debug event occurs during monitor debug-mode debugging, instead of halting the processor, the processor takes an exception. Special software can then take control to examine or alter the processor state. When execution of a monitor target starts, the state of the processor is preserved in the same manner as all ARM exceptions.

15.1.2.1.3 Programming the Debug Unit

The processor unit is programmed using the APB slave port. Features that can be accessed using the memory-mapped APB registers are as follows:

- Instruction address comparators for triggering break points
- Data address comparators for triggering watchpoints
- A bidirectional Debug Communication Channel (DCC)
- All other state information associated with the debug unit

15.1.2.2 ARM Platform ETM

The ETM provides real time instruction trace for the Cortex A8N processor. It is designed to be used with the CoreSight Design Kit. Unlike previous versions of the ARM platforms, this ETM is embedded inside the Core.

Real time tracing is controlled by specifying a set of filtering and triggering resources which include address and data comparators, counters, and sequencers. Note that although data trace cannot be enabled, the ETM can still trigger based on data values. Also, the ETM can trace data address values.

The ETM contains the following main components:

- Core interface
- Trace generation
- Filtering and triggering resources
- Main FIFO
- AMBA 3 ATB interface
- AMBA 3 APB interface

The ETM provides a number of configurations. Table 15-1 shows the options implemented in the ETM.

Table 15-1. ETMv3.3 Configurations

Resource Description	Configuration
Instruction trace	Yes
Data address trace	Yes
Data value trace	No
Jazelle trace	-
Address comparator pairs	2
Data comparator	4
Context ID comparators	1
Sequencer	Yes
Start/stop block	Yes
Embedded ICE comparators	0
External inputs	4
External outputs	2
Extended external inputs	49
Extended external input selectors	2
Instrumentation resources	4
FIFO full	No
FIFO full level setting	N/A
Branch broadcasting	Yes

Resource Description	Configuration
ASIC control register (bits)	8
Data suppression	Yes
Software access to registers	Memory
Readable registers	Yes
FIFO size	128 bytes
Minimum port size	32
Maximum port size	32
Port modes	Dynamic
Asynchronous ATB	Yes
Load pc first	No
Fetch comparisons	No

15.1.2.3 CSETB

The ETB provides on chip storage of trace data. The ARM platform implements a 32-bit 4K RAM.

The ETB accepts trace data from a CoreSight replicator via an ATB write port. The ETB is configurable through an APB port.

Features are as follows:

- 4-Kbyte compiled memory for the trace buffer. Note: It can no longer be used as a general purpose memory.
- Only 32-bit accesses to the 4-Kbyte buffer is supported (Note: RAM size is 4K “bytes”— i.e. $1K \times 32$)
- Amba Peripheral Bus interface for configuration and memory access

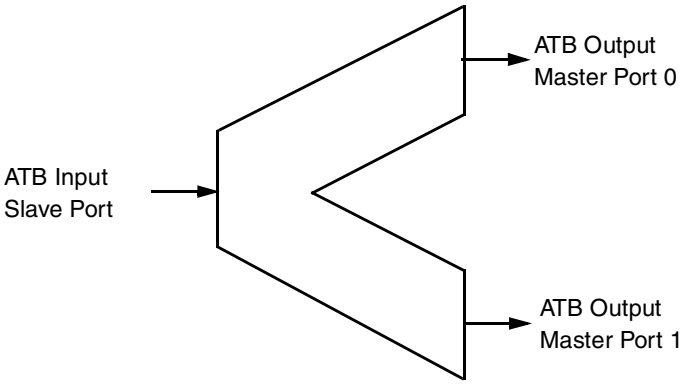
15.1.2.4 CSREPLICATOR

The ATB Replicator enables two trace sinks (TPIU and ETB) to be wired together and receive ATB trace data from the same trace source (ETM). There are no programmable registers. It takes incoming trace data from a single source (ETM) and replicates it to two master ports.

The CSREPLICATOR is part of the tigerp_gp_debug block. Two output master ports are required for this design: one for the on-platform ETB and one for the off-platform TPIU. Placing the TPIU off platform

allows future debug trace sources to connect to the TPIU via a FUNNEL, without the need to modify the ARM platform. Figure 15-2 shows a block diagram for the CSREPLICATOR.

Figure 15-2. CSREPLICATOR Block Diagram



15.1.2.5 CSTPIU

The TPIU acts as a bridge between on-chip trace data, ID distinguishable, and a TPA. It receives ATB trace data and sends it off chip via ARM's standard trace signals TRACECLK, TRACEDATA, and TRACECTL. The following submodules are included in the TPIU (see Figure 15-3): ATB interface, APB interface, Formatter, Asynchronous FIFO, Register bank, Trace out serializer, and a Pattern generator. Further information on the TPIU can be found in ARM's technical reference manuals.

The TPIU is not on the ARM platform and is configurable via the APB.

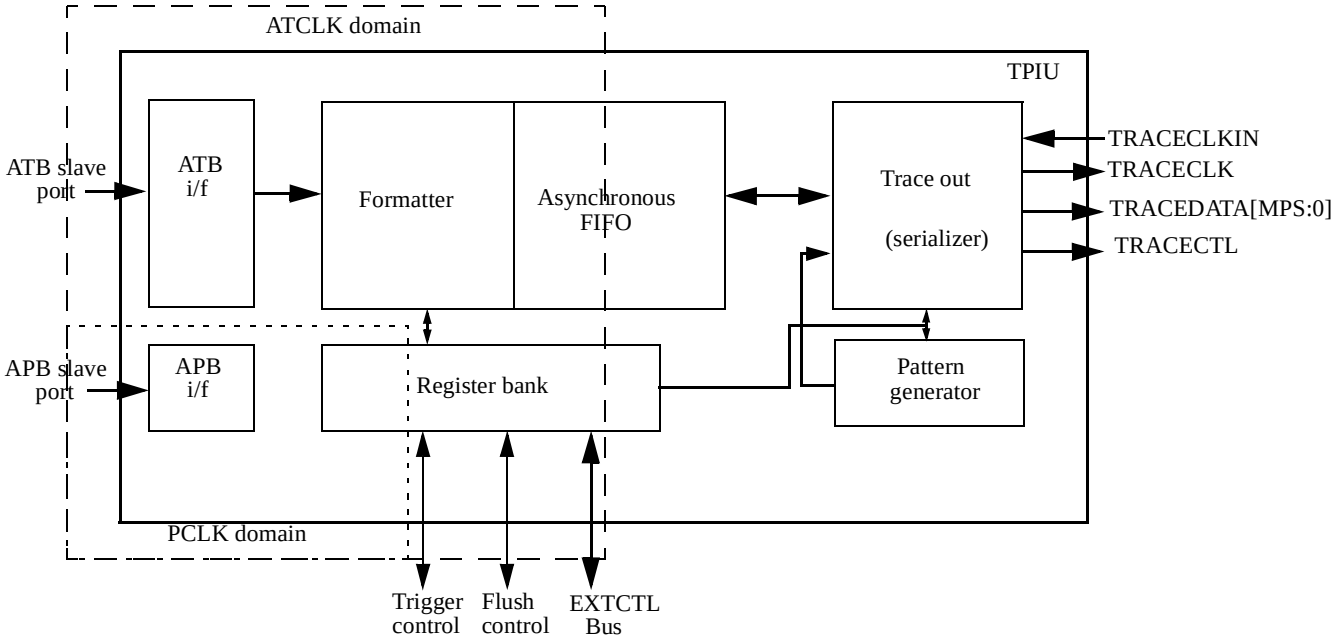


Figure 15-3. TPIU Block Diagram

All signal paths to the pads are subject to wire delays. Special consideration should be taken to re-balance the paths, removing the relative skews between the signals. An extra delay must be added on the TRACECLK path to ensure its rising and falling edge are during the stable part of TRACEDATA and TRACECTL.

15.1.2.6 CSCTI

The CSCTI is the CoreSight Cross Trigger Interface component of a ECT (Embedded Cross Trigger) system. The CSCTI combines and maps trigger requests and broadcast them to all other interfaces on the ECT as channel events. This enables subsystems to cross trigger with each other.

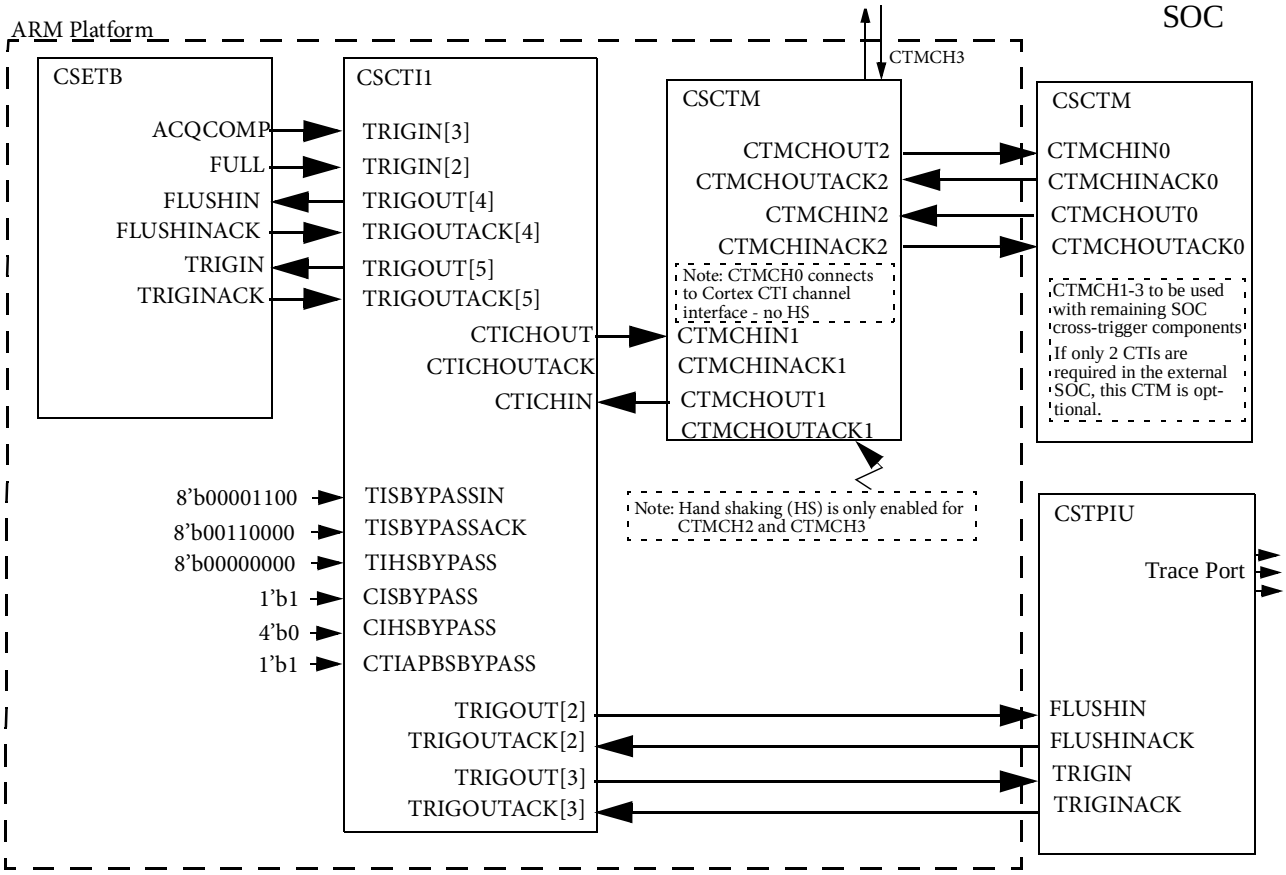
Each CTI has eight trigger inputs, eight trigger outputs, and four sets of 4-channel I/O(s). There are also acknowledge signals, configurable on/off, for the trigger and channel outputs. The channels connect to a CSCTM (CoreSight Cross Trigger Matrix). The CSCTM is off platform so special consideration should be taken for enabling the synchronizers and handshaking features on both the CSCTI and the CSCTM.

The platform requires one CTI module in addition to the embedded CTI in the ARM core. CTI0 resides in the Core and handles triggers/events from the CSETM and the CORE. Where as CTI1 handles the trigger/events for the TPIU (off platform) and the CSETB. Because the TPIU is outside the platform (asynchronous), the TRIGIN synchronizers are enabled and handshaking is turned on.

The on-platform CTI1 is synchronous and at the same clock speed as the on-platform CTM. This allows the channel handshaking and channel synchronizers to be bypassed.

Figure 15-4 shows the recommended connections for CSCTI1

Figure 15-4. CSCTI1 Connections



The SOC CTIs require additional logic to support asynchronous trigger source/destinations. The cscti_extended module includes logic, in addition to the CSCTI module, to support triggering across asynchronous boundaries where the source/destination might not have the necessary logic to support asynchronous boundaries.

15.1.2.7 CSCTM

This block controls the distribution of channel events. It provides channel interfaces to the CSCTIs. The CSCTM can also connect to another CSCTM via a channel interface. This allows multiple CSCTMs to be connected.

The platform has one CSCTM inside the platform to handle events between the platform's CSCTIs and the rest of the ECT system outside the platform. Placing a CSCTM on platform minimizes the event cycle time between the ETB and the ETM. The handshaking between the CTIs on-platform and the on-platform CTM are not enabled. This requires the Cortex's CTI, the on-platform CTI1 and the on-platform CTM to all be in the same clock domain. The on-platform CTM connects to the ECT system via an off-platform CTM and the respective channels require synchronizing and handshaking enabled.

If only two CTIs are required in the SOC, they can be connected directly to the on-platform CTM eliminating the need for a CTM outside the platform.

More information on the CoreSight ECT system can be found in ARM's Technical Reference manuals.

15.1.2.8 DAP

The DAP provides multiple master driving ports, all accessible via a single external interface port. Components that access the DAP are called Debug Ports (DP) and components that access internal interfaces are called Access Ports (AP).

The DAP provides real-time access for the debugger without halting the core to the following:

- All debug configuration registers
- AMBA system memory and peripheral registers.

The DAP enables debug access to the complete SOC via a number of access ports. Access to the CoreSight Debug APB is enabled through the APB access port (APB-AP). System access can be accomplished via an AHB access port (AHB-AP). An APB multiplexor allows system accesses to debug CoreSight components connected to the APB.

There is also a JTAG access port (JTAG-AP) providing accesses to on-chip JTAG components. The DAP acts as a JTAG master. This feature is not used for the ARM platform because the core's debug logic is now accessible via the APB.

Figure 15-5 shows a block diagram of the DAP.

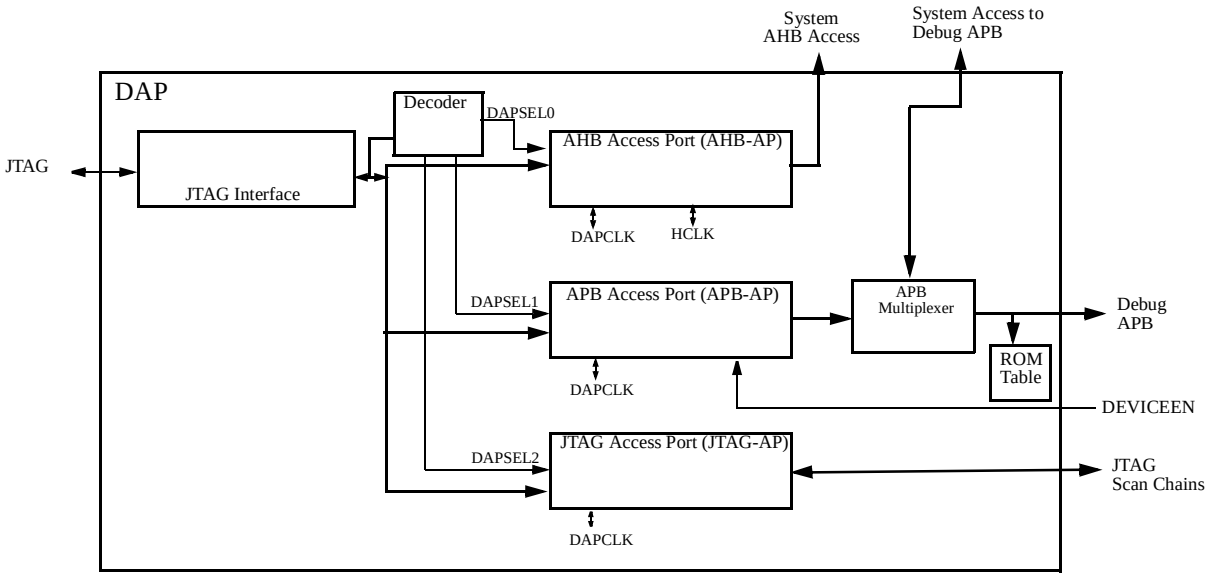


Figure 15-5. DAP Block Diagram

15.1.2.8.1 DAP_SYS

The DAP also requires decode logic for the CoreSight component PSEL signals, an AHB to APB bridge, an AHB lite to AHB bridge, detect logic to enable debug and reset synchronization logic. The dap_sys module wraps these components with the DAP. Figure 15-6 shows a block diagram of the DAP_SYS.

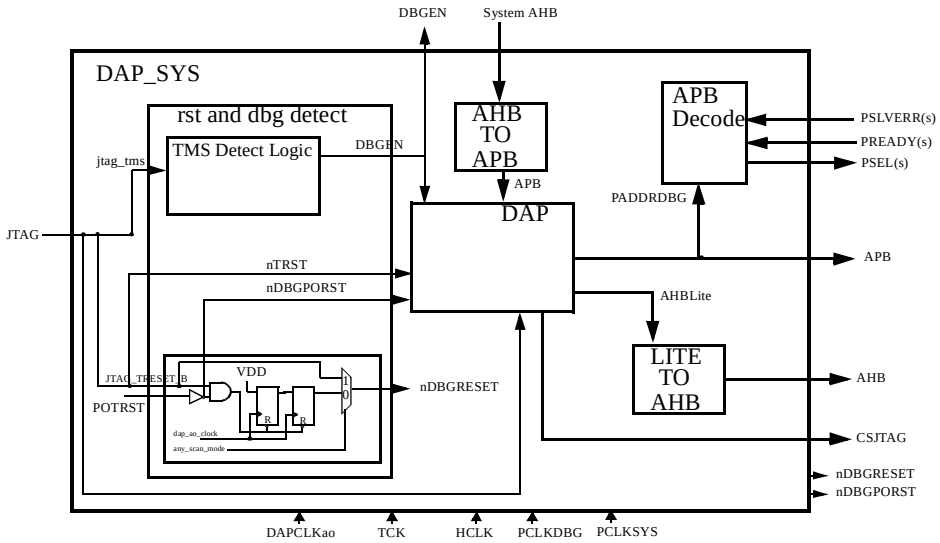


Figure 15-6. DAP_SYS Block Diagram

15.1.3 Modes of Operation

15.1.3.1 ARM Invasive Debug Mode

ARM Invasive Debug mode is usually entered via the JTAG port. Inside the dap_sys there is logic to assert DBGEN high when jtag_tms is active. The user can also invoke DBGEN by writing to a register inside the tiger plat_ctrl module.

It is required that NIDEN is asserted while DBGEN is asserted. Otherwise, CTI functionality is limited when TINIDENSEL is tied to 0. Due to this requirement, NIDEN into the Core and the Debug block is a combination of dbg_niden ored with dbgen_in.

15.1.3.2 ARM Non-Invasive Debug Mode (Real-Time Trace)

There are two methods to enter non-invasive debug mode: through JTAG or via memory mapped accesses. For memory mapped accesses, non-invasive debug can be enabled by writing to two secure supervisor registers, one within the CSU and one within the ARM platform.

15.1.3.3 Normal-Operating Modes

During normal operating mode, debug is not enabled.

15.1.3.4 Low-Power Modes

During deep sleep mode (DSM) all debug is disabled. The platform_ctl module has a bit allocated to overriding DSM when debug is enabled. This bit currently resets to a state disabling DSM while debug is enabled. For more information, refer to the platform control block guide.

The core has an input, DBGNOCLKSTOP, allowing the core's debug clocks to stay on while in WFI. The same bit used to override DSM should be connected to DBGNOCLKSTOP (inverted).

15.2 Memory Map and Register Definition

Each CoreSight component has an allocated 4K. CoreSight components are part of either CoreSight Class or ROM Class. Figure shows the CoreSight class layout for a 4-Kbyte block.

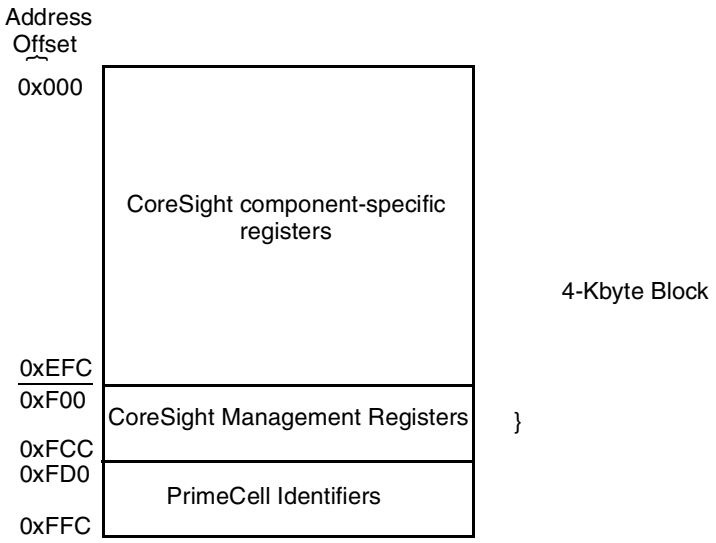


Figure 15-7. CoreSight Component Memory Map

Table 15-2 shows an example Debug memory map. The most significant half word of the address changes based on the chip-level memory map.

Table 15-2. Block Memory Map

Start Address	End Address	Size	Target IP	Comments
B0C00000	B0C00FFF	4K	Debug ROM	Located inside DAP
B0C01000	B0C01FFF	4K	ETB	—
B0C02000	B0C02FFF	4K	ETM	—
B0C03000	B0C03FFF	4K	TPIU	—
B0C04000	B0C04FFF	4K	CTI0	On platform
B0C05000	B0C05FFF	4K	CTI1	On platform

Table 15-2. Block Memory Map (continued)

Start Address	End Address	Size	Target IP	Comments
B0C06000	B0C06FFF	4K	CTI2	Off platform
B0C07000	B0C07FFF	4K	CTI3	Off platform

15.2.1 Register Summary

This section summarizes the CoreSight Component registers in a table format. The tables are divided by Components. For a more detailed description of each register, refer to the CoreSight_DK_TRM.

The conventions in Table 15-3 serve as a key for the register summary.

Table 15-3. Register Conventions

Register Field Types	
RO	Read only. Writing this bit has no effect.
WO	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
RAZ	Read-as-zero
WI	Writes ignored
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)

15.2.1.1 DAP JTAG DP Registers

Accessing a JTAG DP register depends on both the instruction register (IR) value of the DAP access and the address field of the DAP access.

Table 15-4 shows a register summary table for the DAP DP. Detailed register descriptions can be found inside the CoreSight_DK_TRM provided by ARM.

Table 15-4. DAP JTAG DP Registers

Address Field	IR Contents	Type	Register Name
[1]	IDCODE	RO	Identification Control Register
0x0	DAPACC	RAZ/WI	Reserved
0x4	DAPACC	R/W[3]	DP Control/Status Register
0x8	DAPACC	R/W	Select Register
0x0	ABORT	WO[2]	DAP Abort Register
0x4-0xC	ABORT	[2]	Reserved

[1] There is no address associated with the ID code register. For more information, see the CoreSight_DK_TRM.

[2] The value read on the abort scan chain is unpredictable. The result of accessing the abort scan chain without the address set to 0x0 is unpredictable.

15.2.1.2 DAP ROM Register Summary

The ROM table stores the locations of components on the debug APB. The DAP ROM is read only and configurable through the DAPROM.v and DapRomDefs.v RTL files. Writes are ignored.

Table 15-5 shows a register summary table for the DAP Rom. Detailed descriptions can be found inside the CoreSight_DK_TRM provided by ARM.

Table 15-5. DAP ROM Registers

Offset	Name	Type	Description
0xFD0	Peripheral ID4	RO	See CoreSight_DK_TRM
0xFD4	Peripheral ID5	RAZ	Reserved
0xFD8	Peripheral ID6	RAZ	Reserved
0xFDC	Peripheral ID7	RAZ	Reserved
0xFE0	Peripheral ID0	RO	See CoreSight_DK_TRM
0xFE4	Peripheral ID1	RO	See CoreSight_DK_TRM
0xFE8	Peripheral ID2	RO	See CoreSight_DK_TRM
0xFEC	Peripheral ID3	RO	See CoreSight_DK_TRM
0xFF0	Component ID0	RO	Set to 0x0D
0xFF4	Component ID1	RO	Set to 0x10
0xFF8	Component ID2	RO	Set to 0x05
0xFFC	Component ID3	RO	Set to 0xB1

15.2.1.3 Processor Debug Unit Register Summary

Most of the debug unit registers are accessible through the APB. There are some registers that can also be accessed through the coprocessor interface CP14.

By default, CP14 registers can be accessed from a non-privileged mode. However, the processor can be programmed to disable user access modes using bit [12] of the Debug Status and Control Register (DSCR). For more information on the DSCR and access to CP14 registers, refer to the CortexA8 TRM document.

15.2.1.3.1 Coprocessor Registers Summary

Table 15-6 shows the valid debug instructions for accessing the debug registers.

Table 15-6. CP14 Debug Registers Summary

Instruction	Mnemonic	Description
MRC p14, 0, <Rd>, c0, c0, 0	DIDR	Debug Identification Register
MRC p14, 0, <Rd>, c1, c0, 0	DRAR	Debug ROM Address Register

Table 15-6. CP14 Debug Registers Summary

Instruction	Mnemonic	Description
MRC p14, 0, <Rd>, c2, c0, 0	DSAR	Debug Self Address Register
MRC p14, 0, <Rd>, c0, c5, 0 STC p14, c5, <addressing mode>	DTRRX	Data Transfer Register - Receive
MCR p14, 0, <Rd>, c0, c5, 0 LDC p14, c5, <addressing mode>	DTRTX	Data Transfer Register - Transmit
MRC p14, 0, <Rd>, c0, c1, 0 MRC p14, 0, PC, c0, c1, 0	DSCR	Debug Status and Control Register

15.2.1.3.2 Memory-Mapped Registers Summary

Table 15-7 shows a complete list of memory mapped registers accessible using the APB slave port.

Table 15-7. Debug Unit APB Accessible Registers Summary

Offset	Register Number	Mnemonic	Type	Description
0x000	c0	DIDR	R	CP14 c0, Debug ID Register
0x004–0x014	c1–c5	—	R	RAZ
0x018	c6	WFAR	R/W	Watchpoint Fault Address Register
0x01C	c7	VCR	R/W	Vector Catch Register
0x020	c8	—	R	RAZ
0x024	c9	ECR	R/W	Event Catch Register
0x028	c10	DSCCR	R/W	Debug State Cache Control Register
0x02C	c11	—	R	RAZ
0x030–0x07C	c12–c31	—	R	RAZ
0x080	c32	DTRRX	R/W	Data Transfer Register—Receive
0x084	c33	ITR	W	Instruction Transfer Register
0x088	c34	DSCR	R/W	CP14 c1, Debug Status and Control Register
0x08C	c35	DTRTX	R/W	Data Transfer Register—Transmit
0x090	c36	DRCR	W	Debug Run Control Register
0x094–0x09C	c37–c63	—	R	RAZ
0x100–0x114	c64–c69	BVR	R/W	Breakpoint Value Registers
0x118–0x13c	c70–c79	—	R	RAZ
0x140–0x154	c80–c85	BCR	R/W	Breakpoint Control Register
0x158–0x17C	c86–c95	—	R	RAZ
0x180–0x184	c96–c97	WVR	R/W	Watchpoint Value Register
0x188–0x1BC	c97–c111	—	R	RAZ
0x1C0–0x1C4	c112–c113	WCR	R/W	Watchpoint Control Registers

Table 15-7. Debug Unit APB Accessible Registers Summary (continued)

Offset	Register Number	Mnemonic	Type	Description
0x1C8-0x1FC	c114-c127	—	R	RAZ
0x200-0x2FC	c128-c191	—	R	RAZ
0x300	c192	OSLAR	W	Operating System Lock Access Register
0x304	c193	OSLSR	R	Operating System Lock Status Register
0x308	c194	OSSRR	R/W	Operating System Save and Restore Register
0x30C	c195	—	R	RAZ
0x310	c196	PRCR	R/W	Device Power Down and Reset Control Register
0x314	c197	PRSR	R	Device Power Down and Reset Status Register
0x318-0x7FC	c198-c511	—	R	RAZ
0x800-0x8FC	c512-575	—	R	RAZ
0x900-0xCFC	c576-c831	—	R	RAZ
0xD00-0xFFC	c832-c1023	—	-	Management Register

15.2.1.4 ETB Register Summary

The ETB registers are summarized in Table 15-8. Detailed information can be found in ARM's CoreSight_DK_TRM document.

Table 15-8. ETB Registers

Offset	Name	Type	Description
0x004	RAM Depth Register, RDP	RO	See CoreSight_DK_TRM
0x010	RAM Read Data Register, RRD	RO	See CoreSight_DK_TRM
0x014	RAM Read Pointer Register, RRP	R/W	See CoreSight_DK_TRM
0x00C	Status Register, STS	RO	See CoreSight_DK_TRM
0x018	RAM Write Pointer Register, RWP	R/W	See CoreSight_DK_TRM
0x01C	Trigger Counter, TRG	R/W	See CoreSight_DK_TRM
0x020	Control Register, CTL	R/W	See CoreSight_DK_TRM
0x024	RAM Write Data, RWD	WO	See CoreSight_DK_TRM
0x300	Formatter and Flush Status FFSR	RO	See CoreSight_DK_TRM
0x304	Formatter and Flush Control FFCR	R/W	See CoreSight_DK_TRM
0xEE0	Integration Register ITMISCOP0	WO	See CoreSight_DK_TRM
0xEE4	Integration Register ITTRFLINACK	WO	See CoreSight_DK_TRM

Table 15-8. ETB Registers (continued)

Offset	Name	Type	Description
0xEE8	Integration Register ITTRFLIN	RO	See CoreSight_DK_TRM
0xEEC	Integration Register ITATBDATA0	RO	See CoreSight_DK_TRM
0xEF0	Integration Register ITATBCTR2	WO	See CoreSight_DK_TRM
0xEF4	Integration Register ITATBCTR1	RO	See CoreSight_DK_TRM
0xEF8	Integration Register ITATBCTR0	RO	See CoreSight_DK_TRM
0xF00	Integration Mode Control Register	R/W	See CoreSight_DK_TRM
0xFA0	Claim Tag Set Register	R/W	See CoreSight_DK_TRM
0xFA4	Claim Tag Clear Register	R/W	See CoreSight_DK_TRM
0xFB0	Lock Access Register	WO	See CoreSight_DK_TRM
0xFB4	Lock Status Register	RO	See CoreSight_DK_TRM
0xFB8	Authentication Status Register	RO	See CoreSight_DK_TRM
0xFC8	Device ID	RO	See CoreSight_DK_TRM
0xFCC	Device Type Identifier Register	RO	See CoreSight_DK_TRM
0xFD0	Peripheral ID4	RO	See CoreSight_DK_TRM
0xFD4	Peripheral ID5	RAZ	Reserved
0xFD8	Peripheral ID6	RAZ	Reserved
0xFDC	Peripheral ID7	RAZ	Reserved
0xFE0	Peripheral ID0	RO	See CoreSight_DK_TRM
0xFE4	Peripheral ID1	RO	See CoreSight_DK_TRM
0xFE8	Peripheral ID2	RO	See CoreSight_DK_TRM
0xFEC	Peripheral ID3	RO	See CoreSight_DK_TRM
0xFF0	Component ID0	RO	Set to 0x0D
0xFF4	Component ID1	RO	Set to 0x10
0xFF8	Component ID2	RO	Set to 0x05
0xFFC	Component ID3	RO	Set to 0xB1

15.2.1.5 ETM Register Summary

The ETM registers are described in Table 15-9. A more detailed description of the ETM registers can be found in ARM's ETM_ARCHITECTURE_SPEC document.

Table 15-9. ETM Register Summary

Offset	Name	Type	Description
0x00	ETM Control	R/W	See ETMv3.3 Architecture Specification
0x04	ETM Configuration Control	RO	See ETMv3.3 Architecture Specification
0x08	Trigger Event	WO[33]	See ETMv3.3 Architecture Specification
0x0C	ASIC Control	WO[3]	See ETMv3.3 Architecture Specification
0x10	ETM Status	R/W	See ETMv3.3 Architecture Specification
0x14	System Configuration	RO	See ETMv3.3 Architecture Specification
0x18	Trace Start/Stop Resource Control	WO[3]	See ETMv3.3 Architecture Specification
0x1C	Trace Enable Control 2	WO[3]	See ETMv3.3 Architecture Specification
0x20	TraceEnable	WO[3]	See ETMv3.3 Architecture Specification
0x24	TraceEnable Control 1	WO[3]	See ETMv3.3 Architecture Specification
0x30	ViewData Event	WO[3]	See ETMv3.3 Architecture Specification
0x34	ViewData Control 1	WO[3]	See ETMv3.3 Architecture Specification
0x38	ViewData Control 2	WO[3]	See ETMv3.3 Architecture Specification
0x3C	ViewData Control 3	WO[3]	See ETMv3.3 Architecture Specification
0x40–0x7C	Address Comparator Value 1–16	WO[3]	See ETMv3.3 Architecture Specification
0x80–0xBC	Address Access Type 1–16	WO[3]	See ETMv3.3 Architecture Specification
0xC0–0xFC	Data Comparator Value 1–16	WO[3]	See ETMv3.3 Architecture Specification
0x40–0x4F	Data Comparator Mask 1–16	WO[3]	See ETMv3.3 Architecture Specification
0x50–0x53	Counter Reload Value 1–4	WO[3]	See ETMv3.3 Architecture Specification
0x54–0x57	Counter Enable 1–4	WO[3]	See ETMv3.3 Architecture Specification
0x58–0x5B	Counter Reload Event 1–4	WO[3]	See ETMv3.3 Architecture Specification
0x5C–0x5F	Counter Value 1–4	R/W	See ETMv3.3 Architecture Specification
0x60–0x65	Sequencer State Transition Event	WO[3]	See ETMv3.3 Architecture Specification
0x66	—	—	Reserved
0x67	Current Sequencer State	R/W	See ETMv3.3 Architecture Specification

Table 15-9. ETM Register Summary (continued)

Offset	Name	Type	Description
0x68–0x6B	External Output Event 1–4	WO[3]	See ETMv3.3 Architecture Specification
0x6C–0x6E	Context ID Comparator Value	WO[3]	See ETMv3.3 Architecture Specification
0x6F	Context ID Comparator Mask	WO[3]	See ETMv3.3 Architecture Specification
0x70–0x77	Implementation specific	WO[3]	See ETMv3.3 Architecture Specification
0x78	Synchronization Frequency	WO[3]	See ETMv3.3 Architecture Specification
0x79	ETM ID	RO	See ETMv3.3 Architecture Specification
0x7A	Configuration Code Extension	RO	See ETMv3.3 Architecture Specification
0x7B	Extended External Input Selection	WO[3]	See ETMv3.3 Architecture Specification
0x7C	Trace Start/Stop Embedded ICE Control	WO[3]	See ETMv3.3 Architecture Specification
0x7D	Embedded ICE Behavior Control	WO[3]	See ETMv3.3 Architecture Specification
0x7E–0x7F	—	—	Reserved
0x080	CoreSight Trace ID	R/W	See ETMv3.3 Architecture Specification
0x81–0xBF	—	—	Reserved
0xC0	OS Lock Access	WO	See ETMv3.3 Architecture Specification
0xC1	OS Lock Status	RO	See ETMv3.3 Architecture Specification
0xC2	OS Save/Restore	R/W	See ETMv3.3 Architecture Specification
0xC3–0xCF	—	—	Reserved
0x380–0x3BF	Integration registers	—	Reserved for Implementation-defined topology detection and integration registers.
0xF00	Integration Mode Control	R/W	See ETMv3.3 Architecture Specification
0xFA0	Claim Tag Set	R/W	See ETMv3.3 Architecture Specification
0xFA4	Claim Tag Clear	R/W	See ETMv3.3 Architecture Specification
0xFB0	Lock Access	WO	See ETMv3.3 Architecture Specification
0xFB4	Lock Status	RO	See ETMv3.3 Architecture Specification
0xFB8	Authentication Status	RO	See ETMv3.3 Architecture Specification

Table 15-9. ETM Register Summary (continued)

Offset	Name	Type	Description
0xFC8	Device Configuration	RO	See ETMv3.3 Architecture Specification
0xFCC	Device Type	RO	See ETMv3.3 Architecture Specification
0xFD0	Peripheral ID4	RO	See ETMv3.3 Architecture Specification
0xFD4	Peripheral ID5	RO	See ETMv3.3 Architecture Specification
0xFD8	Peripheral ID6	RO	See ETMv3.3 Architecture Specification
0xFDC	Peripheral ID7	RO	See ETMv3.3 Architecture Specification
0xFE0	Peripheral ID0	RO	See ETMv3.3 Architecture Specification
0xFE4	Peripheral ID1	RO	See ETMv3.3 Architecture Specification
0xFE8	Peripheral ID2	RO	See ETMv3.3 Architecture Specification
0xFEC	Peripheral ID3	RO	See ETMv3.3 Architecture Specification
0xFF0	Component ID0	RO	See ETMv3.3 Architecture Specification
0xFF4	Component ID1	RO	See ETMv3.3 Architecture Specification
0xFF8	Component ID2	RO	See ETMv3.3 Architecture Specification
0xFFC	Component ID3	RO	See ETMv3.3 Architecture Specification

[3] In ETMv3.1 and later, register is read/write if bit [11] of the ETM Configuration Code Extension Register (0x7A) is set to b1.

15.2.1.6 CSCTI Register Summary

The CSCTI registers are described in Table 15-10. A more detailed description of the CSCTI registers can be found in ARM's CoreSight_DK_TRM document.

Table 15-10 describes the location by an offset.

Table 15-10. CSCTI Register Summary

Offset	Name	Type	Width	Reset Value	Description
0x000	CTICONTROL	R/W	1	0x0	See ARM CoreSight_DK_TRM
0x010	CTIINTACK	WO	8	—	See ARM CoreSight_DK_TRM
0x014	CTIAPPSET	R/W	4	0x0	See ARM CoreSight_DK_TRM
0x018	CTIAPPCLEAR	WO	?	0x0	See ARM CoreSight_DK_TRM
0x01C	CTIAPPPULSE	WO	4	0x0	See ARM CoreSight_DK_TRM
0x020–0x03C	CTIINEN	R/W	4	0x00	See ARM CoreSight_DK_TRM

Table 15-10. CSCTI Register Summary (continued)

Offset	Name	Type	Width	Reset Value	Description
0x0A0–0x0BC	CTIOUTEN	R/W	4	0x00	See ARM CoreSight_DK_TRM
0x130	CTITRIGINSTATUS	RO	8	—	See ARM CoreSight_DK_TRM
0x134	CTITRIGOUTSTATUS	RO	8	0x00	See ARM CoreSight_DK_TRM
0x138	CTICHINSTATUS	RO	4	—	See ARM CoreSight_DK_TRM
0x13C	CTICHOUTSTATUS	RO	4	0x0	See ARM CoreSight_DK_TRM
0x140	Channel gate	R/W	4	0xF	See ARM CoreSight_DK_TRM
0x144	External multiplexor control	R/W	8	0x00	See ARM CoreSight_DK_TRM
0xEDC	ITCHINACK	WO	4	0x0	See ARM CoreSight_DK_TRM
0xEE0	ITTRIGINACK	WO	8	0x00	See ARM CoreSight_DK_TRM
0xEE4	ITCHOUT	WO	4	0x0	See ARM CoreSight_DK_TRM
0xEE8	ITTRIGOUT	WO	8	0x00	See ARM CoreSight_DK_TRM
0xEEC	ITCHOUTACK	RO	4	0x0	See ARM CoreSight_DK_TRM
0xEF0	ITTRIGOUTACK	RO	8	0x00	See ARM CoreSight_DK_TRM
0xEF4	ITCHIN	RO	4	0x0	See ARM CoreSight_DK_TRM
0xEF8	ITTRIGIN	RO	8	0x00	See ARM CoreSight_DK_TRM
0xEFC–0xF7C	—	—	—	—	See ARM CoreSight_DK_TRM
0xF00	ITCTRL	R/W	1	0x0	See ARM CoreSight_DK_TRM
0xFA0	Claim Tag Set	R/W	4	0xF	See ARM CoreSight_DK_TRM
0xFA4	Claim Tag Clear	R/W	4	0x0	See ARM CoreSight_DK_TRM
0xFB0	Lock Access Register	WO	32	—	See ARM CoreSight_DK_TRM

Table 15-10. CSCTI Register Summary (continued)

Offset	Name	Type	Width	Reset Value	Description
0xFB4	Lock Status Register	RO	2	0x3	See ARM CoreSight_DK_TRM
0xFB8	Authentication Status	RO	4	0xA	See ARM CoreSight_DK_TRM
0xFC0–0xFC4	—	—	—	—	See ARM CoreSight_DK_TRM
0xFC8	Device ID	RO	20	0x40800	See ARM CoreSight_DK_TRM
0xFCC	Device Type Identifier	RO	8	0x14	See ARM CoreSight_DK_TRM
0xFD0	PeripheralID4	RO	8	0x04	See ARM CoreSight_DK_TRM
0xFD4	PeripheralID5	—	—	—	See ARM CoreSight_DK_TRM
0xFD8	PeripheralID6	—	—	—	See ARM CoreSight_DK_TRM
0xFDC	PeripheralID7	—	—	—	See ARM CoreSight_DK_TRM
0xFE0	PeripheralID0	RO	8	0x06	See ARM CoreSight_DK_TRM
0xFE4	PeripheralID1	RO	8	0xB9	See ARM CoreSight_DK_TRM
0xFE8	PeripheralID2	RO	8	0x0B	See ARM CoreSight_DK_TRM
0xFEC	PeripheralID3	RO	8	0x00	See ARM CoreSight_DK_TRM
0xFF0	Component ID0	RO	8	0x0D	See ARM CoreSight_DK_TRM
0xFF4	Component ID1	RO	8	0x90	See ARM CoreSight_DK_TRM
0xFF8	Component ID2	RO	8	0x05	See ARM CoreSight_DK_TRM
0xFFC	Component ID3	RO	8	0xB1	See ARM CoreSight_DK_TRM

15.2.1.7 TPIU Register Summary

The TPIU (Trace Port Interface Unit) registers are described in Table 15-11. A more detailed description of the TPIU registers can be found in ARM's CoreSight_DK_TRM document. These registers can be accessed via the APB port and are memory map accessible.

Table 15-11. TPIU Register Summary

Offset	Name	Type	Width	Reset value	Description
0x000	Supported port sizes	RO	32	0xFFFF_FFFF	See ARM CoreSight_DK_TRM
0x004	Current port size	R/W	32	0x0000_0001	See ARM CoreSight_DK_TRM
0x100	Supported trigger modes	RO	18	0x11F	See ARM CoreSight_DK_TRM
0x104	Trigger counter value	R/W	8	0x00	See ARM CoreSight_DK_TRM
0x108	Trigger multiplier	R/W	5	0x00	See ARM CoreSight_DK_TRM
0x200	Supported test pattern/modes	RO	18	0x3000F	See ARM CoreSight_DK_TRM
0x204	Current test pattern/mode	RO	18	0x00000	See ARM CoreSight_DK_TRM
0x208	Test pattern repeat counter	R/W	8	0x00	See ARM CoreSight_DK_TRM
0x300	Formatter flush and status	RO	3	0x6	See ARM CoreSight_DK_TRM
0x304	Fomatter flush and control	R/W	14	0x1000	See ARM CoreSight_DK_TRM
0x308	Formatter synchronization counter	R/W	12	0x040	See ARM CoreSight_DK_TRM
0x400	EXTCTL In Port	RO	8	Undefined	See ARM CoreSight_DK_TRM
0x408	EXTCTL Out Port	R/W	8	0x00	See ARM CoreSight_DK_TRM
0xEE4	Integration Register, ITTRFLINACK	WO	2	—	See ARM CoreSight_DK_TRM
0xEE8	Integration Register, ITTRFLIN	WO	2	Undefined	See ARM CoreSight_DK_TRM
0xEEC	Integration Register, ITATBDATA0	WO	5	Undefined	See ARM CoreSight_DK_TRM
0xEF0	Integration Register, ITATBCTR2	WO	2	—	See ARM CoreSight_DK_TRM
0xEF4	Integration Register, ITATBCTR1	RO	7	Undefined	See ARM CoreSight_DK_TRM

Table 15-11. TPIU Register Summary (continued)

Offset	Name	Type	Width	Reset value	Description
0xEF8	Integration Register, ITATBCTR0	RO	10	Undefined	See ARM CoreSight_DK_TRM
0xF00	Integration Register, ITATBCTR0	R/W	1	0x0	See ARM CoreSight_DK_TRM
0xFA0	Claim Tag Set Register	R/W	4	0xF	See ARM CoreSight_DK_TRM
0xFA4	Claim Tag Clear Register	R/W	4	0x0	See ARM CoreSight_DK_TRM
0xFB0	Lock Access Register	WO	32	—	See ARM CoreSight_DK_TRM
0xFB4	Lock Status Register	RO	3	0x3	See ARM CoreSight_DK_TRM
0xFB8	Authentication Status Register	RO	8	0x00	See ARM CoreSight_DK_TRM
0xFC8	Device ID	RO	32	0x00	See ARM CoreSight_DK_TRM
0xFCC	Device Type Identifier Register	RO	8	0x21	See ARM CoreSight_DK_TRM
0xFD0	Peripheral ID4	RO	8	0x04	See ARM CoreSight_DK_TRM
0xFD4	Peripheral ID5	RO	8	0x00 (reserved)	See ARM CoreSight_DK_TRM
0xFD*	Peripheral ID6	RO	8	0x00 (reserved)	See ARM CoreSight_DK_TRM
0xFDC	Peripheral ID7	RO	8	0x00 (reserved)	See ARM CoreSight_DK_TRM
0xFE0	Peripheral ID0	RO	8	0x07	See ARM CoreSight_DK_TRM
0xFE4	Peripheral ID1	RO	8	0xB9	See ARM CoreSight_DK_TRM
0xFE8	Peripheral ID2	RO	8	0x0B	See ARM CoreSight_DK_TRM
0xFEC	Peripheral ID3	RO	8	0x00	See ARM CoreSight_DK_TRM
0xFF0	Component ID0	RO	8	0x0D	See ARM CoreSight_DK_TRM
0xFF4	Component ID1	RO	8	0x90	See ARM CoreSight_DK_TRM

Table 15-11. TPIU Register Summary (continued)

Offset	Name	Type	Width	Reset value	Description
0xFF8	Component ID2	RO	8	0x05	See ARM CoreSight_DK_TRM
0xFFC	Component ID3	RO	8	0xB1	See ARM CoreSight_DK_TRM

15.2.2 Clocks

The list below describes the Debug clocks within the ARM platform.

- ATCLK

  This is the AMBA Trace Bus (ATB) clock. This clock is also used to clock the on-platform CTICLK and on-platform CTMCLK. ATCLK needs to be synchronous (equivalent or faster) to PCLKDBG.
- CTICLK

  This is the main clock for the CTI module. The current platform configuration requires this clock to be synchronous to the on-platform CTM clock.
- PCLKDBG

  This is the Debug APB clock. It must be synchronous (equivalent or slower) to the ATCLK.
- TRACECLKIN

  This is the Trace Port Interface Unit trace clock input. Since the trace port pads are limited to 133 MHz, TRACECLKIN is pad limited to less than or equal to 266 MHz.
- TRACECLK

  This is the clock for the external trace port. It is a DDR clock and runs at ½ the speed of TRACECLKIN.

The platform control module generates the debug clocks from the arm clock. The requirement above states that the PCLKDBG needs to be synchronous to the ATCLK. When configuring the divide by settings, make sure this requirement is followed. PCLKDBG is a divide by two of ATCLK.

15.2.3 Reset

All debug resets are derived from JTAG_TRST_B and POR. These two resets go into the DAP_SYS to be combined and de-asserted synchronously to the DAPCLKao (always on clock). They also go into the platform, being synchronously de-asserted inside the plat_cntrl module, generating resets for the debug logic inside the platform. Debug resets that do not require synchronous de-assertion are routed directly to

their destination, such as por for the Core and ETM. Figure 15-8 shows a diagram of the platform debug reset strategy.

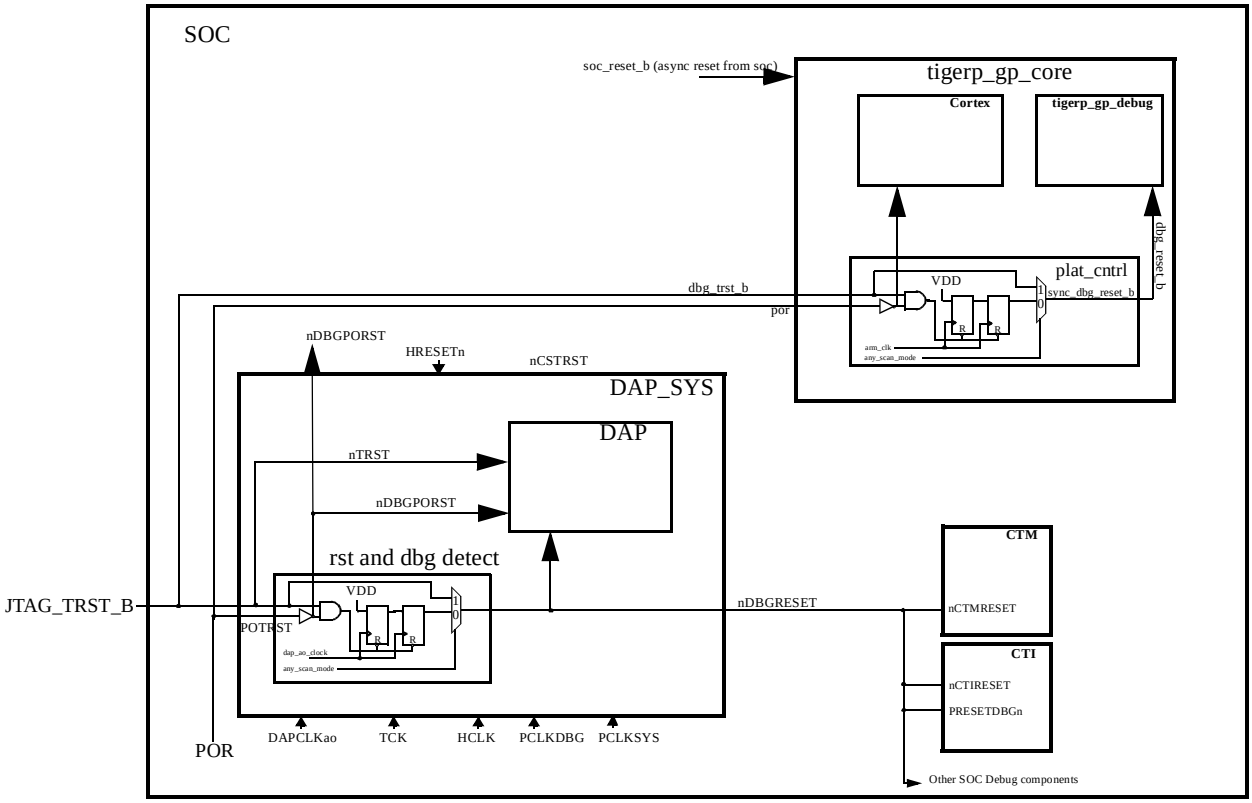


Figure 15-8. Platform Debug Reset Strategy

15.2.4 Endianness

Little endianness is supported; big endianness is not supported.

Chapter 16
Multi-Layer AHB Crossbar Switch (MAX)

This chapter provides an overview of the MAX (Multi-Layer AHB Crossbar Switch). The purpose of the MAX is to concurrently support up to 4 simultaneous connections between master ports and slave ports. The MAX supports a 32-bit address bus width and a 32-bit data bus width at all master and slave ports. A simplified block diagram is shown in Figure 16-1.

NOTE

The MAX implements a 7 master by 4 slave configuration.

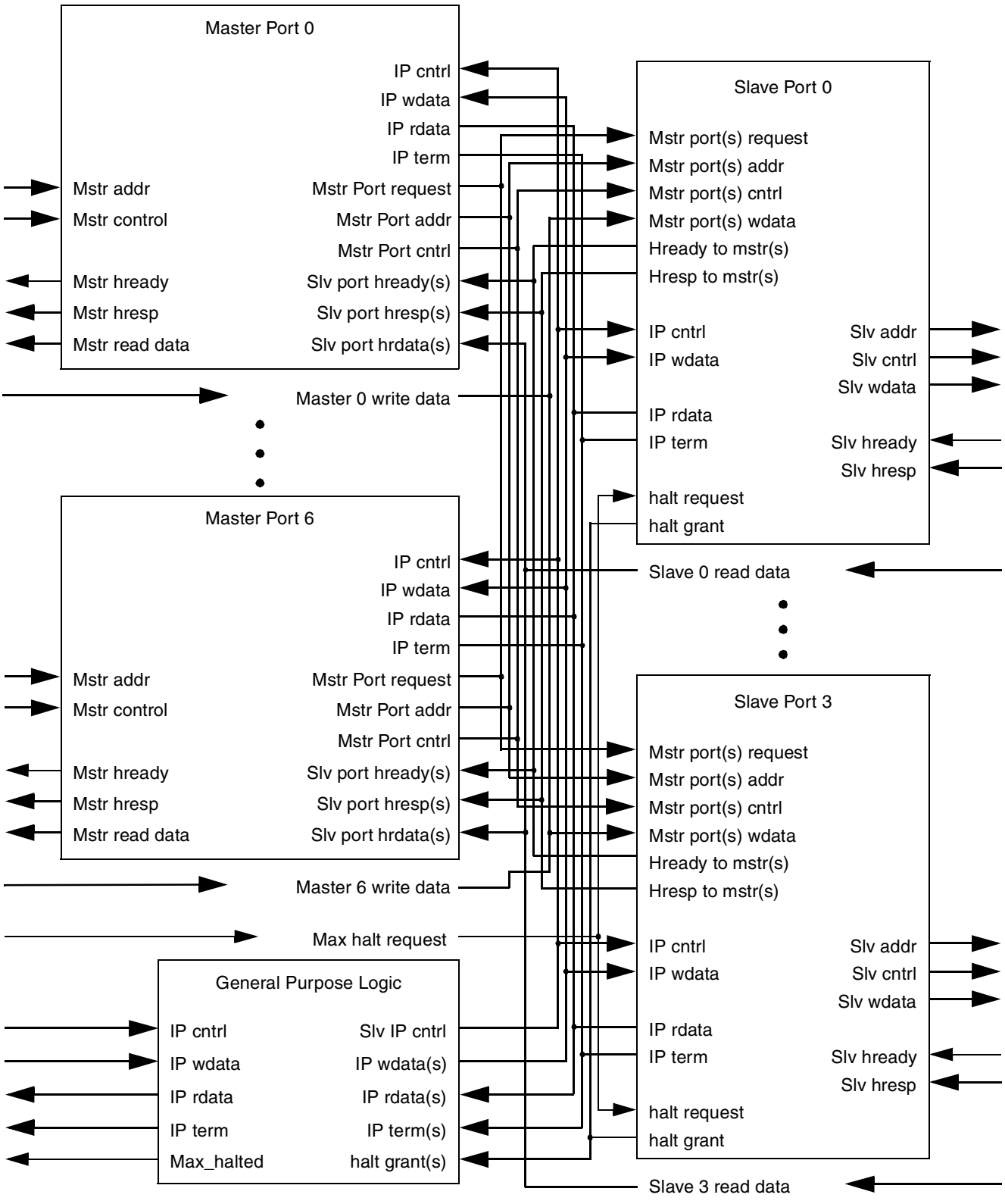


Figure 16-1. MAX Block Diagram

16.1 Features

The MAX has the ability to gain control of all the slave ports and prevent any masters from making accesses to the slave ports. This feature is useful when the user wishes to turn off the clocks to the system and needs to ensure that no bus activity is interrupted.

The MAX can put each slave port into a low-power park mode so that the slave port does not dissipate any power transitioning address, control, or data signals when not being actively accessed by a master port.

Each slave port can also support multiple master priority schemes. Each slave port has a hardware input which selects the master priority scheme so the user can dynamically change master priority levels on a slave port by slave port basis.

The MAX allows concurrent transactions to occur from any master port to any slave port. It is possible for four master ports and all slave ports to be in use at the same time as a result of independent master requests. If a slave port is simultaneously requested by more than one master port, arbitration logic selects the higher priority master and grant it ownership of the slave port. All other masters requesting that slave port are stalled until the higher priority master completes its transactions.

16.1.1 Limitations

The MAX routes bus transactions initiated on the master ports to the appropriate slave ports. There is no provision included to route transactions initiated on the slave ports to other slave ports or to master ports. Simply put, the slave ports do not support the bus request/bus grant protocol, the MAX assumes it is the sole master of each slave port.

Because the MAX does not support the bus request/bus grant protocol, an external arbiter needs to be used if multiple masters are to be connected to a single master port. In the case of a single master connecting to a master port, the single master's bus grant signal must be tied off in the asserted state.

Each master and slave port is fully AHB-Lite + AMBA V6 extensions compliant. The ports are not fully AHB compliant because the MAX does not support SPLITs or RETRYs.

16.1.2 General Operation

When a master makes an access to the MAX, the access is immediately taken by the MAX. If the targeted slave port of the access is available, the access is immediately presented on the slave port. It is possible to make single clock (zero wait state) accesses through the MAX. If the targeted slave port of the access is busy or parked on a different master port, the requesting master simply sees wait states inserted (**hready** held negated) until the targeted slave port can service the master's request. The latency in servicing the request depends on each master's priority level and the responding peripheral's access time.

Since the MAX appears to be just another slave to the master device, the master device has no knowledge of whether or not it actually owns the slave port it is targeting. While the master does not have control of the slave port it is targeting, it is simply wait stated.

A master is given control of the targeted slave port only after a previous access to a different slave port has completed, regardless of its priority on the newly targeted slave port. This prevents deadlock from occurring when a master has an outstanding request to one slave port that has a long response time, has a

pending access to a different slave port, and a lower priority master is also making a request to the same slave port as the pending access of the higher priority master.

Once the master has control of the slave port it is targeting the master remains in control of that slave port until it gives up the slave port by running an IDLE cycle or by leaving that slave port for its next access. The master could also lose control of the slave port if another higher priority master makes a request to the slave port; however, if the master is running a locked or fixed length burst transfer it retains control of the slave port until that transfer is completed. Based on the AULB bit in the MGPCR (Master General Purpose Control Register) the master either retains control of the slave port when doing undefined length incrementing burst transfers or loses the bus to a higher priority master.

The MAX terminates all master IDLE transfers (as opposed to allowing the termination to come from one of the slave busses). Additionally, when no master is requesting access to a slave port the MAX drives IDLE transfers onto the slave bus, even though a default master may be granted access to the slave port. When the MAX is controlling the slave bus (i.e. during low power park or halt mode) the **hmaster** field indicates 4'b0000.

When a slave bus is being IDLEd by the MAX, it can park the slave port on the master port indicated by the PARK bits in the SGPCR (Slave General Purpose Control Register). This can be done in an attempt to save the initial clock of arbitration delay that would otherwise be seen if the master had to arbitrate to gain control of the slave port. The slave port can also be put into low power park mode in attempt to save power.

16.2 MAX Interface Signals

This section provides information on MAX interface signals, including AHB master and slave interface signals as well as IP bus interface signals.

16.2.1 MAX Signal Descriptions

Please reference the AMBA Specification Rev 2.0 for a description of the AHB signals in the MAX and the IP Bus Specification Rev 2.0 for a description of the IP Bus signals in the MAX.

16.2.1.1 max_halt_request

This input signal is a request to halt all slave port bus activity (run MAX originated IDLE cycles on each slave port bus, blocking all master port accesses). This signal can be used to gracefully shut down the MAX so the system clock can be stopped for low power mode. This signal is captured by a flop inside the MAX before use.

Once the MAX is halted, it remains halted until **max_halt_request** is negated.

16.2.1.2 max_halted

This output is asserted once the MAX is in control and running IDLE cycles on each slave port.

16.3 Memory Map and Register Definition

There are four registers that reside in each slave port of the MAX and one register that resides in each master port of the MAX. These registers are IP bus compliant registers. Read and write transfers both require two IP bus clock cycles. The registers can only be read from and written to in supervisor mode. Additionally, these registers can only be read from or written to by 32-bit accesses. Section 16.3.3, MAX Register Descriptions," provides the detailed descriptions for all of the MAX registers.

The registers are fully decoded and an error response is returned if an unimplemented location is accessed within the MAX.

The slave registers also feature a bit, which when written with a 1 prevents the registers from being written to again. The registers are still readable, but future write attempts have no effect on the registers and will be terminated with an error response.

16.3.1 Memory Map

Table 16-1 shows the MAX memory map.

Table 16-1. MAX Memory Map

Offset	Register	Access	Reset Value	Section/Page
General Registers				
0x43F0_4000(MPR0)	Master Priority Register for Slave port 0	R/W	0x0054_3210	16.3.3.1/16-9
0x43F0_4010 (SGPCR0)	General Purpose Control Register for Slave port 0	RW	0x000_000	16.3.3.2/16-10
0x43F0_4100(MPR1)	Master Priority Register for Slave port 1	R/W	0x0054_3210	16.3.3.1/16-9
0x43F0_4110 (SGPCR1)	General Purpose Control Register for Slave port 1	RW	0x000_000	16.3.3.2/16-10
0x43F0_4200(MPR2)	Master Priority Register for Slave port 2	R/W	0x0054_3210	16.3.3.1/16-9
0x43F0_4210 (SGPCR2)	General Purpose Control Register for Slave port 2	RW	0x000_000	16.3.3.2/16-10
0x43F0_4300(MPR3)	Master Priority Register for Slave port 3	R/W	0x0054_3210	16.3.3.1/16-9
0x43F0_4310 (SGPCR3)	General Purpose Control Register for Slave port 3	RW	0x000_000	16.3.3.2/16-10
0x43F0_4800 (MGPCR0)	General Purpose Control Register for Master port 0	R/W	0x0000_0000	16.3.3.3/16-12
0x43F0_4900 (MGPCR1)	General Purpose Control Register for Master port 1	R/W	0x0000_0000	16.3.3.3/16-12
0x43F0_4A00 (MGPCR2)	General Purpose Control Register for Master port 2	R/W	0x0000_0000	16.3.3.3/16-12

Table 16-1. MAX Memory Map (continued)

Offset	Register	Access	Reset Value	Section/Page
0x43F0_4B00 (MGPCR3)	General Purpose Control Register for Master port 3	R/W	0x0000_0000	16.3.3.3/16-12
0x43F0_4C00 (MGPCR4)	General Purpose Control Register for Master port 4	R/W	0x0000_0000	16.3.3.3/16-12
0x43F0_4D00 (MGPCR5)	General Purpose Control Register for Master port 5	R/W	0x0000_0000	16.3.3.3/16-12
0x43F0_4E00 (MGPCR6)	General Purpose Control Register for Master port 6	R/W	0x0000_0000	16.3.3.3/16-12

16.3.2 Register Summary

Figure 16-2 shows the key to the register fields and Table 16-2 shows the register figure conventions.

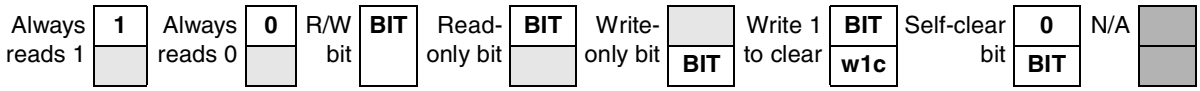


Figure 16-2. Key to Register Fields

Table 16-2. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 16-3 shows the MAX register summary.

Table 16-3. MAX Detailed Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43F0_4000(MPR0)	R	0	0	0	0	0	MSTR_6			0	MSTR_5			0	MSTR_4		
	W																
	R	0	MSTR_3			0	MSTR_2			0	MSTR_1			0	MSTR_0		
	W																
0x43F0_4100(MPR1)	R	0	0	0	0	0	MSTR_6			0	MSTR_5			0	MSTR_4		
	W																
	R	0	MSTR_3			0	MSTR_2			0	MSTR_1			0	MSTR_0		
	W																
0x43F0_4200(MPR2)	R	0	0	0	0	0	MSTR_6			0	MSTR_5			0	MSTR_4		
	W																
	R	0	MSTR_3			0	MSTR_2			0	MSTR_1			0	MSTR_0		
	W																
0x43F0_4300(MPR3)	R	0	0	0	0	0	MSTR_6			0	MSTR_5			0	MSTR_4		
	W																
	R	0	MSTR_3			0	MSTR_2			0	MSTR_1			0	MSTR_0		
	W																
	W																
0x43F0_4010 (SGPCR0)	R	RO	HLP	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	ARB		0	0	PCTL		0	PARK		
	W																
0x43F0_4110 (SGPCR1)	R	RO	HLP	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	ARB		0	0	PCTL		0	PARK		
	W																
0x43F0_4210 (SGPCR2)	R	RO	HLP	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	ARB		0	0	PCTL		0	PARK		
	W																
0x43F0_4310 (SGPCR3)	R	RO	HLP	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	ARB		0	0	PCTL		0	PARK		
	W																
0x43F0_4800 (MGPCR0)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																

Table 16-3. MAX Detailed Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43F0_4900 (MGPCR1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																
0x43F0_4A00 (MGPCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																
0x43F0_4B00 (MGPCR3)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																
0x43F0_4C00 (MGPCR4)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																
0x43F0_4D00 (MGPCR5)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																
0x43F0_4E00 (MGPCR6)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
	W																

16.3.3 MAX Register Descriptions

This section contains the detailed register descriptions for the MAX registers.

16.3.3.1 Master Priority Register (MPR0-MPR3)

The Master Priority Register (MPR) sets the priority of each master port on a per slave port basis and resides in each slave port. See Figure 16-3 for illustration of valid bits in the MPR and Table 16-4 for description of the bit fields.

0x43F0_4000(MPR0)
0x43F0_4100(MPR1)
0x43F0_4200(MPR2)
0x43F0_4300(MPR3)

Access: Supervisor read-write

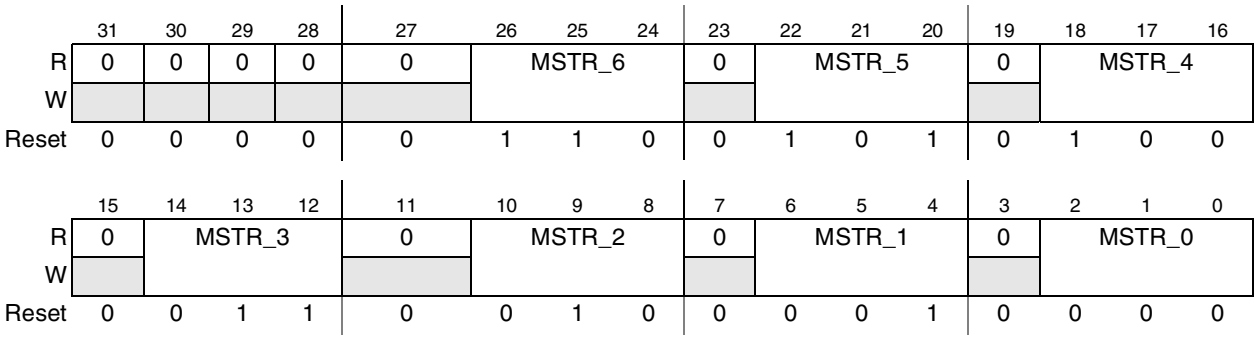


Figure 16-3. Master Priority Register (MPR0 - MPR3)

Table 16-4. Master Priority Register Descriptions

Field	Description
31–27	Reserved. They are read as zero and should be written with zero for upward compatibility.
26–24 MSTR_6	Master 6Priority. These bits set the arbitration priority for master port 6 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
23	Reserved. They are read as zero and should be written with zero for upward compatibility.
22–20 MSTR_5	Master 5 Priority. These bits set the arbitration priority for master port 5 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
19	Reserved. They are read as zero and should be written with zero for upward compatibility.
18–16 MSTR_4	Master 4 Priority. These bits set the arbitration priority for master port 4 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
15	Reserved. They are read as zero and should be written with zero for upward compatibility.

Table 16-4. Master Priority Register Descriptions (continued)

Field	Description
14–12 MSTR_3	Master 3 Priority. These bits set the arbitration priority for master port 3 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
11	Reserved. They are read as zero and should be written with zero for upward compatibility.
10–8 MSTR_2	Master 2 Priority. These bits set the arbitration priority for master port 2 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
7	Reserved. They are read as zero and should be written with zero for upward compatibility.
6–4 MSTR_1	Master 1 Priority. These bits set the arbitration priority for master port 1 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111 This master has the lowest priority when accessing the slave port.
3	Reserved. They are read as zero and should be written with zero for upward compatibility.
2–0 MSTR_0	Master 0 Priority. These bits set the arbitration priority for master port 0 on the associated slave port. These bits are initialized by hardware reset. 000 This master has the highest priority when accessing the slave port. 111This master has the lowest priority when accessing the slave port.

The Master Priority Register can only be accessed in supervisor mode with 32-bit accesses. Once the RO (Read Only) bit has been set in the Slave General Purpose Control Register, the Master Priority Register can only be read from. Attempts to write to it have no effect on the MPR and result in an error response.

Additionally, no two available master ports may be programmed with the same priority level. Attempts to program two or more available masters with the same priority level results in an error response, and the MPR is not updated.

16.3.3.2 Slave General Purpose Control Register (SGPCR0-SGPCR3)

The Slave General Purpose Control Register (SGPCR) controls several features of each slave port.

The Read Only (RO) bit prevents any registers associated with this slave port from being written to once set. This bit may be written with 0 as many times as the user desires, but once it is written to a 1 only a reset condition allows it to be written again.

The Halt Low Priority (HLP) bit sets the priority of the **max_halt_request** input to the lowest possible priority for initial arbitration of the slave ports. By default it is the highest priority. Please note, setting this bit does not affect the **max_halt_request** from attaining highest priority once it has control of the slave ports.

The PCTL bits determine how the slave port parks when no master is actively making a request. The available options are to park on the master defined by the PARK bits, park on the last master to use the slave port, or go into a low-power park mode that forces all the outputs of the slave port to inactive states when no master is requesting an access. The low-power park feature can result in an overall power savings

if a the slave port is not saturated; however, it forces an extra clock of latency whenever any master tries to access it when it is not in use because it is not parked on any master.

The PARK bits determine which master the slave parks on when no master is making an active request and the **max_halt_request** input is negated. Please use caution to only select master ports that are actually present in the design. If the user programs the PARK bits to a master not present in the current design implementation, undefined behavior results.

See Figure 16-4 for illustration of valid bits in the SGPCR, and Table 16-5 for description of the bit fields.

0x43F0_4010 (SGPCR0)
0x43F0_4110 (SGPCR1)
0x43F0_4210 (SGPCR2)
0x43F0_4310 (SGPCR3)

Access: Supervisor read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RO	HLP	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	ARB		0	0	PCTL		0	PARK		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 16-4. Slave General Purpose Control Register *n*

Table 16-5. Slave General Purpose Control Register Descriptions

Field	Description
31 RO	Read Only This bit is used to force all of a slave port's registers to be read only. Once written to 1 it can only be cleared by hardware reset. This bit is initialized by hardware reset. 0 All this slave port's registers can be written. 1 All this slave port's registers are read only and cannot be written (attempted writes have no effect and result in an error response).
30 HLP	Halt Low Priority This bit is used to set the initial arbitration priority of the max_halt_request input. This bit is initialized by hardware reset. 0 The max_halt_request input has the highest priority for arbitration on this slave port 1 The max_halt_request input has the lowest initial priority for arbitration on this slave port.
29–10	Reserved. They read as zero and should be written with zero for upward compatibility.
9–8 ARB	Arbitration Mode These bits are used to select the arbitration policy for the slave port. These bits are initialized by hardware reset. 00 Fixed Priority. 01 Round Robin (rotating) Priority. 10 Reserved 11 Reserved
7–6	Reserved. They read as zero and should be written with zero for upward compatibility.

Table 16-5. Slave General Purpose Control Register Descriptions (continued)

Field	Description
5–4 PCTL	Parking Control These bits determine the parking control used by this slave port. These bits are initialized by hardware reset. 00 When no master is making a request, the arbiter parks the slave port on the master port defined by the PARK bit field. 01 When no master is making a request, the arbiter parks the slave port on the last master to be in control of the slave port. 10 When no master is making a request, the arbiter parks the slave port on no master and will drive all outputs to a constant safe state. 11 Reserved
3	Reserved. They read as zero and should be written with zero for upward compatibility.
2–0 PARK	PARK These bits are used to determine which master port this slave port parks on when no masters are actively making requests and the PCTL bits are set to 00. These bits are initialized by hardware reset. 000 Park on Master Port 0 001 Park on Master Port 1 010 Park on Master Port 2 011 Park on Master Port 3 100 Park on Master Port 4 101 Park on Master Port 5 110 Park on Master Port 6 111 Reserved

The SGPCR can only be accessed in supervisor mode with 32-bit accesses. Once the RO (Read Only) bit has been set in the SGPCR the SGPCR can only be read, attempts to write to it have no effect on the SGPCR and result in an error response.

16.3.3.3 Master General Purpose Control Register

The Master General Purpose Control Register (MGPCR) presently controls only whether or not the master's undefined length burst accesses are allowed to complete uninterrupted or whether they can be broken by requests from higher priority masters.

The AULB (Arbitrate on Undefined Length Bursts) bit field determines whether (and when) the MAX arbitrates away the slave port the master owns when the master is performing undefined length burst accesses.

0x43F0_4800 (MGPCR0)
0x43F0_4900 (MGPCR1)
0x43F0_4A00 (MGPCR2)
0x43F0_4B00 (MGPCR3)
0x43F0_4C00 (MGPCR4)
0x43F0_4D00 (MGPCR5)
0x43F0_4E00 (MGPCR6)

Access: Supervisor read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	AULB		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 16-5. Master General Purpose Control Register *n*

Table 16-6. Master General Purpose Control Register Descriptions

Name	Description
31 - 3	Reserved. They read as zero and should be written with zero for upward compatibility.
2 - 0 AULB	Arbitrate on Undefined Length Bursts. These bits are used to select the arbitration policy during undefined length bursts by this master. These bits are initialized by hardware reset. 000 No arbitration is allowed during an undefined length burst. 001 Arbitration is allowed at any time during an undefined length burst. 010 Arbitration is allowed after four beats of an undefined length burst. 011 Arbitration is allowed after eight beats of an undefined length burst. 100 Arbitration is allowed after 16 beats of an undefined length burst. 101 Reserved 110 Reserved 111 Reserved

The MGPCR can only be accessed in supervisor mode with 32-bit accesses.

16.3.4 Coherency

Since the content of the registers has a real time effect on the operation of the MAX it is important for the user to understand that any register modifications take effect as soon as the register is written. The values of the registers do not track with slave port related AHB accesses but instead track only with IP bus accesses.

The exception to this rule are the AULB bits in the MGPCR. The update of these bits is only recognized when the master on that master port runs an IDLE cycle, even though the IP bus cycle to write them will have long since terminated successfully. If the AULB bits in the MGPCR are written in between two burst

accesses, the new AULB encodings does not take effect until an IDLE cycle has been initiated by the master on that master port.

16.4 Detailed Functional Description

This section describes the functionality of the MAX in greater detail.

16.4.1 Arbitration

The MAX supports two arbitration schemes; a simple fixed-priority comparison algorithm, and a simple round-robin fairness algorithm. The arbitration scheme is independently programmable for each slave port.

16.4.1.1 Arbitration During Undefined Length Bursts

Arbitration points during an undefined length burst are defined by the current master's MGPCR AULB field setting. When a defined length is imposed on the burst via the AULB bits, the undefined length burst is treated as a single or series of single back to back fixed length burst accesses. For example, a master runs an undefined length burst and the AULB bits in the MGPCR indicate arbitration occurs after the fourth beat of the burst. The master runs two sequential beats and then starts a 12-beat, undefined length burst access to a new address within the same slave port region as the previous access. The MAX does not allow an arbitration point until the fourth overall access (second beat of the second burst). At that point all remaining accesses are open for arbitration until the master loses control of the slave port.

Assume the master loses control of the slave port after the fifth beat of the second burst. Once the master regains control of the slave port, no arbitration point will be available until after the master has run four more beats of its burst. After the fourth beat of the (now continued) burst (ninth beat of the second burst from the master's perspective) is taken, all beats of the burst will once again be open for arbitration until the master loses control of the slave port.

Assume the master again loses control of the slave port on the fifth beat of the third (now continued) burst (10th beat of the second burst from the master's perspective). Once the master regains control of the slave port, it will be allowed to complete its final two beats of its burst without facing arbitration.

Note that fixed-length burst accesses arenot be affected by the AULB bits. All fixed- length burst accesses lock out arbitration until the last beat of the fixed-length burst.

16.4.1.2 Fixed Priority Operation

When operating in fixed-priority mode, each master is assigned a unique priority level in the MPR (Master Priority Register). If two masters both request access to a slave port, the master with the highest priority in the selected priority register gains control over the slave port.

Any time a master makes a request to a slave port, the slave port checks to see if the new requesting master's priority level is higher than that of the master that currently has control over the slave port (unless the slave port is in a parked state). The slave port does an arbitration check at every clock edge to ensure that the proper master (if any) has control of the slave port.

If the new requesting master's priority level is higher than that of the master that currently has control of the slave port the new requesting master is granted control over the slave port at the next clock edge. The exception to this rule is if the master that currently has control over the slave port is running a fixed length burst transfer or a locked transfer. In this case the new requesting master must wait until the end of the burst transfer or locked transfer before it will be granted control of the slave port. If the master is running an undefined length burst transfer the new requesting master must wait until an arbitration point for the undefined length burst transfer before it will be granted control of the slave port. Arbitration points for an undefined length burst are defined in the MGPCR for each master.

If the new requesting master's priority level is lower than that of the master that currently has control of the slave port, the new requesting master is forced to wait until the master that currently has control of the slave port either runs an IDLE cycle or runs a non IDLE cycle to a location other than the current slave port.

16.4.1.3 Round-Robin Priority Operation

When operating in round-robin mode, each master is assigned a relative priority based on the master number. This relative priority is compared to the ID of the last master to perform a transfer on the slave bus. The highest priority requesting master becomes owner of the slave bus as the next transfer boundary (accounting for locked and fixed-length burst transfers). Priority is based on how far ahead the ID of the requesting master is to the ID of the last master (ID is defined by master port number, not the **hmaster** field).

Once granted access to a slave port, a master may perform as many transfers as desired to that port until another master makes a request to the same slave port. The next master in line is granted access to the slave port at the next assertion of **sX_hready**, or possibly on the next clock cycle if the current master has no pending access request.

As an example of arbitration in round-robin mode, assume the MAX is implemented with master ports 0, 1, 2, 3, 4 and 5. If the last master of the slave port was master 1, and master 0, 4 and 5 make simultaneous requests, (master ports 2 and 3 make no requests), they are serviced in the order 4, 5, and then 0.

Parking may still be used in a round-robin mode, but does not affect the round-robin pointer unless the parked master actually performs a transfer. Handoff to the next master in line occurs after one cycle of arbitration. If the slave port is put into low-power park mode, the round-robin pointer is reset to point at master port 0, giving it the highest priority.

16.4.2 Priority Assignment

Each master port needs to be assigned a unique 3 bit priority level. If an attempt is made to program multiple master ports with the same priority level within a register (MPR), the MAX responds with an error and the registers are not updated.

16.4.3 Master Port Functionality

This section discusses the master port functionality.

16.4.3.1 General

Each master port consists of two decoders, a capture unit, a register slice, a mux, and a small state machine.

The first decoder is used to decode the **haddr** and control signals coming directly from the master, telling the state machine where the master's next access will be and if it is in fact a legal access. The second decoder gets its input from the capture unit, so it may be looking directly at the signals coming from the master or it may be looking at captured signals coming from the master, depending entirely on the state of the targeted slave port. The second decoder is then used to generate the access requests that go to the slave ports.

The capture unit is used to capture the address and control information coming from the master in the event that the targeted slave port cannot immediately service the master. The capture unit is controlled by outputs from the state machine which tell it to either pass through the original master signals or the captured signals.

The register slice contains the registers associated with the specific master port. The registers have a quasi-IP bus interface at this level for reads and writes and the outputs feed directly into the state machine.

The mux is used simply to select which slave's read data is sent back to the master. The mux is controlled by the state machine.

The state machine controls all aspects of the master port. It knows which slave port the master wants to make a request to and controls when that request is made. It also has knowledge of each slave port, knowing whether or not the slave port is ready to accept an access from the master port. This determines whether or not the master may immediately have its request taken by the slave port or whether the master port must capture the master's request and queue it at the slave port boundary.

For a block diagram of a master port see Figure 16-6.

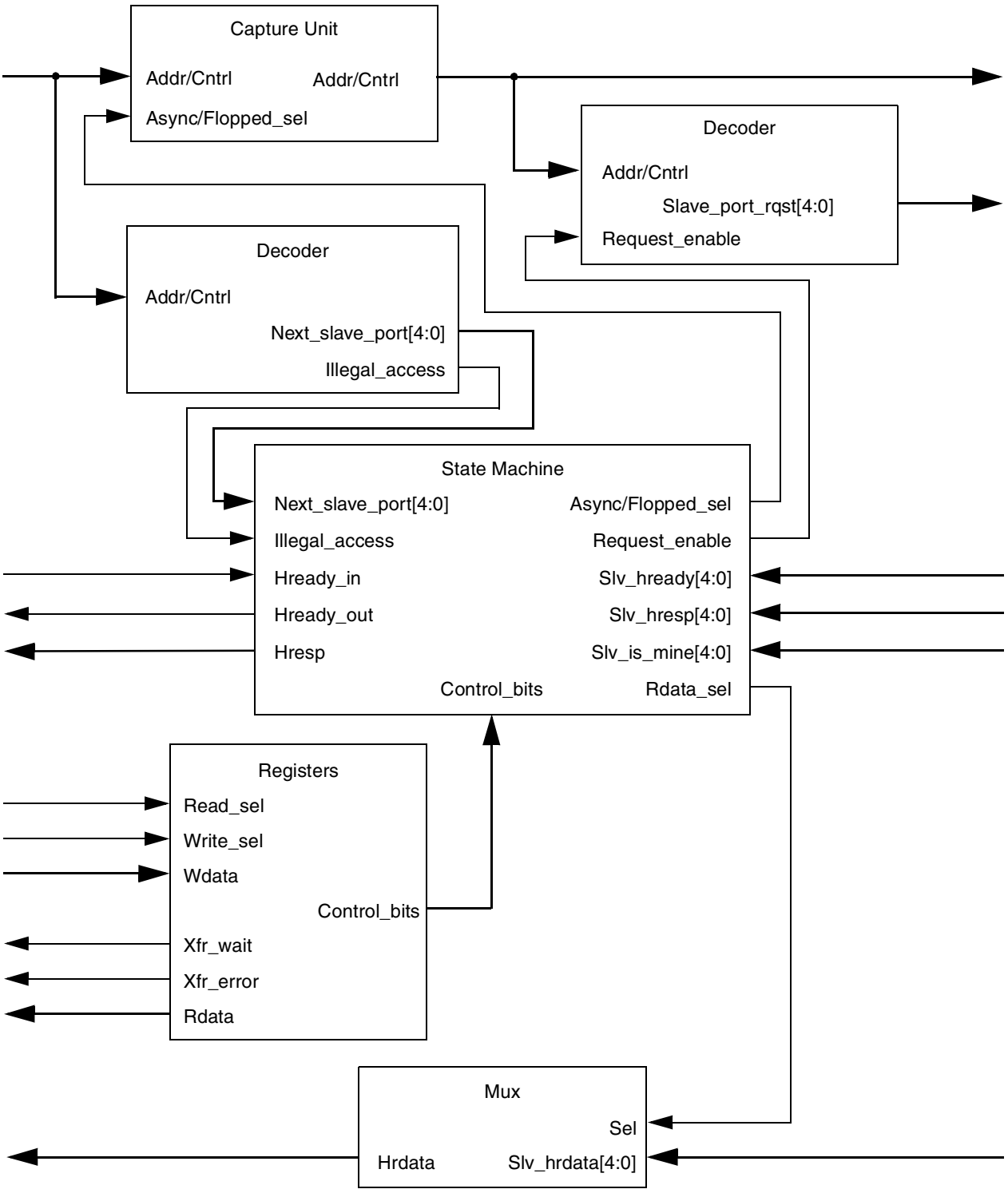


Figure 16-6. MAX Master Port Block Diagram

16.4.3.2 Master Port Decoders

The decoders are very simple as they ensure an access request is allowed to be made and that the slave port targeted is actually present in the design. The decoders feeding the state machine are always enabled. The decoders that select the slave are enabled only when the master port controlling state machine wants to make a request to a slave port. This is necessary so that if a master port is making an access to a slave port and is being wait stated, and its next access is to a different slave port, the request to the second slave port can be held off until the access to the first slave port is terminated.

The decoders also output a “hole decode” or illegal access signal which tells the state machine that the master is trying to access a slave port that does not exist.

16.4.3.3 Master Port Capture Unit

The capture unit simply captures the state of the master’s address and control signals if the MAX cannot immediately pass the master’s request through to the proper slave port. The capture unit consists of a set of flops and a mux which selects either the asynchronous path from address and control or the flopped (captured) address and control information.

16.4.3.4 Master Port Registers

The registers in the master port are only those registers associated with this particular master port. The read and write interface for the registers is a quasi-IP bus interface. It is not a full IP bus interface at this level because not all the IP bus signals are routed this deep in the design.

There is a register control block at the same level of the master port and slave port instantiations in the MAX. This control block ensures that all accesses are 32-bit supervisor accesses before passing them on to the master ports.

The register outputs are connected directly to the state machine.

16.4.3.5 Master Port State Machine

This section describes the master port state machine states and state swapping.

16.4.3.5.1 Master Port State Machine States

The master side state machine’s main function is to monitor the activities of the master port. The state machine has six states: **busy**, **idle**, **stalled**, **steady state**, **first cycle error response** and **second cycle error response**.

The **busy** state is used when the master runs a BUSY cycle to the master port. The master port maintains its request to the slave port if it currently owns the slave port; however, if it loses control of the slave port it no longer maintains its request. If the master port loses control of the slave port, it is not allowed to make another request to the slave port until it runs a NSEQ or SEQ cycle.

The **idle** state is used when the master runs a valid IDLE cycle to the master port. The master port makes no requests to the slave ports (disables the slave port decoder) and terminates the IDLE cycle.

The **stalled** state is used when the master makes a request to a slave port that is not immediately ready to receive the request. In this case the state machine directs the capture unit to send out the captured address and control signals and enables the slave port decoder to indicate a pending request to the appropriate slave port.

The **steady state** is used when the master port and slave port are in fully asynchronous mode, making the MAX completely transparent in the access. The state machine selects the appropriate slave's **hresp0, hready** and **hrdata** to pass back to the master.

The **first cycle error response** and **second cycle error response** states are self explanatory. The MAX responds with an error response to the master if the master tries to access an unimplemented memory location through the MAX (i.e. a slave port that does not exist).

16.4.3.5.2 Master Port State Machine Slave Swapping

The design of the master side state machine is fairly straight forward. The one real decision to be made is how to handle the master moving from one slave port access to another slave port access. The approach that was taken was to minimize or eliminate, when possible, any bubbles that would get inserted into the access due to switching slave ports.

The state machine does not allow the master to request access to another slave port until the current access being made is terminated. This prevents a single master from owning two slave ports at the same time (the slave port it is currently accessing and the slave port it wishes to access next).

The state machine also maintains watch on the slave port the master is accessing as well as the slave port the master wishes to switch to. If the new slave port is parked on the master, the master is able to make the switch without incurring any delays. The termination of the current access also acts as the launch of the new access on the new slave port. If the new slave port is not parked on the master, the master incurs a minimum one-clock delay before it can launch its access on the new slave port.

This is the same for switching from the **busy** or **idle** state to actively accessing a slave port. If the slave port is parked on the master, the state machine goes to the **steady state** and the access begins immediately. If the slave port is not parked on the master (serving another master, parked on another master or in low power park mode), the state machine transitions to the **stalled** state and at least a one-clock penalty is paid.

16.4.4 Slave Port Functionality

This section discusses the slave port functionality.

16.4.4.1 General

Each slave port consists of a register slice, a bank of muxes and a state machine.

The register slice contains the registers associated with the specific slave port. The registers have a quasi-IP bus interface at this level for reads and writes and the outputs feed directly into the state machine.

The muxes are a series of 6 to 1 muxes that take in all the address, control and write data information from each of the master ports and then pass the correct master's signals to the slave port. The state machine controls all the muxes.

The state machine is where the main slave port arbitration occurs, it decides which master is in control of the slave port and which master will be in control of the slave port in the next bus cycle.

For a block diagram of a slave port see Figure 16-7.

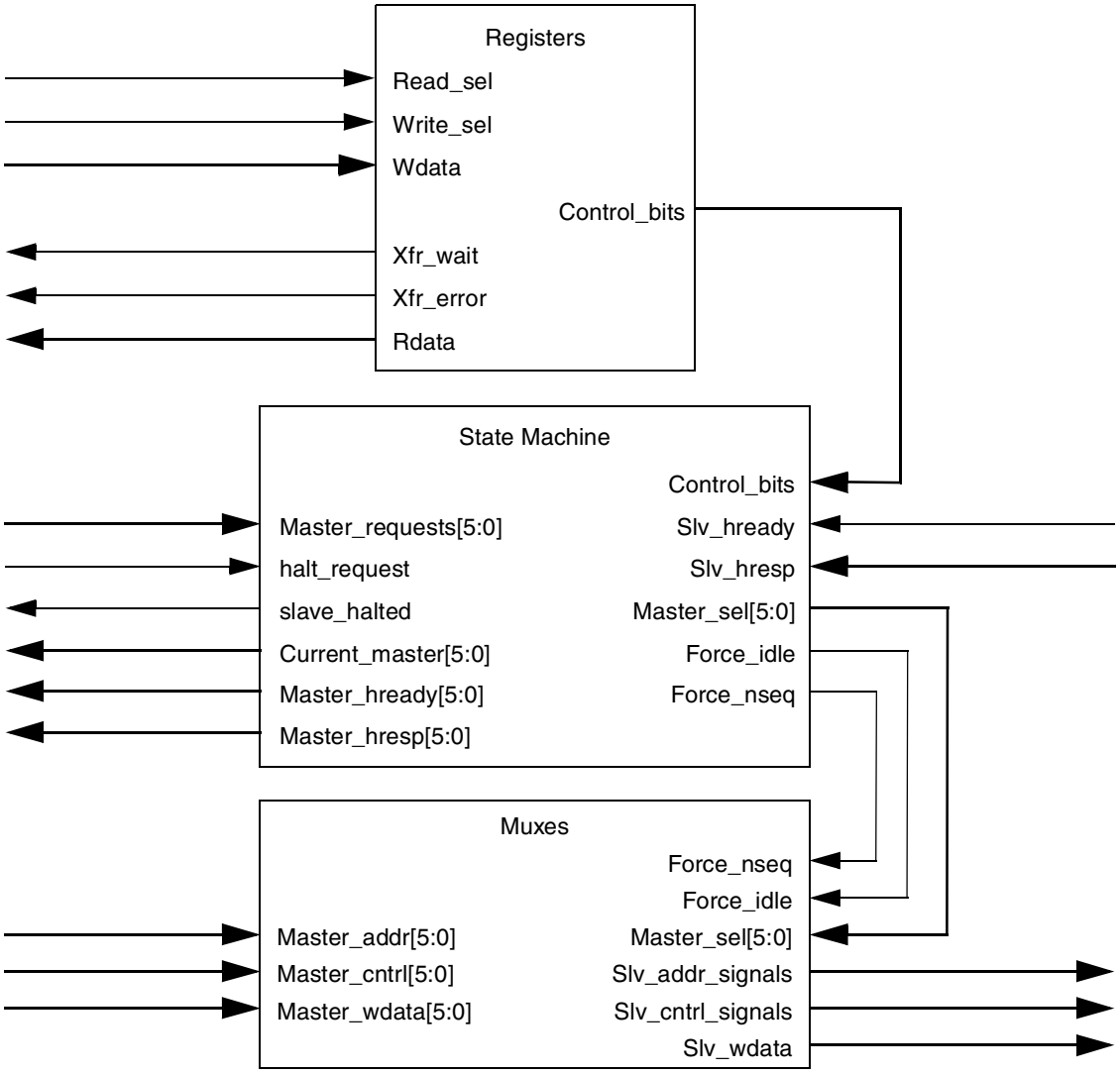


Figure 16-7. MAX Slave Port Block Diagram

16.4.4.2 Slave Port Muxes

The block diagram (Figure 16-7) shows only one block for all the muxes. In reality that block instantiates many 6 to 1 muxes, one for each master-to-slave signal in fact. All the muxes are designed in an AND-OR fashion, so that if no master is selected, the output of the muxes is zero. (This is an important feature for low-power park mode.)

The muxes also have an override signal which is used by the slave port to asynchronously force IDLE cycles onto the slave bus. When the state machine forces an IDLE cycle it zeros out **htrans** and **hmastlock**, making sure the slave bus sees a valid IDLE cycle being run by the MAX.

The enable to the mux controlling **htrans** also contains an additional control signal from the state machine so that a NSEQ transaction can be forced. This is done any time the slave port switches masters to ensure that no IDLE-SEQ, BUSY-SEQ or NSEQ-SEQ transactions are seen on the slave port when they should not be. If the state machine indicates to run both an IDLE and an NSEQ cycle, the IDLE directive has priority.

NOTE

IDLE-SEQ is in fact an illegal access, but a possible scenario given the multimaster environment in the MAX unless corrected by the MAX.

16.4.4.3 Slave Port Registers

There is a register control block at the same level of the master port and slave port instantiations in the MAX. This control block ensures that all accesses are 32-bit supervisor accesses before passing them on to the master and slave ports.

The registers in the slave port are only those registers associated with this particular slave port. The read and write interface for the registers is a quasi-IP bus interface. It is not a full IP bus interface at this level because not all the IP bus signals are routed this deep in the design.

The register outputs are connected directly to the slave state machine. The registers can be read from an unlimited number of times. The registers can only be written to as long as the RO bit is written to 0 in the SGPCR, once it is written to a 1 only a hardware reset allows the registers to be written again.

16.4.4.4 Slave Port State Machine

This section discusses the slave port state machine states, arbitration, master handoff, machine parking, and halt mode.

16.4.4.4.1 Slave Port State Machine States

At the heart of the slave port is the state machine. The state machine is simplicity itself, requiring only four states - **steady state**, **transition state, transition hold state,** and **hold state**. Either the slave port is owned by the same master it was in the last clock cycle (either by active use or by parking), it is transitioning to a new master (either for active use or parking), it is transitioning to a new master during wait states or it is being held on the same master pending a transition to a new master.

16.4.4.4.2 Slave Port State Machine Arbitration

The real work in the state machine is determining which master port will be in control of the slave port in the next clock cycle, the arbitration. Each master is programmed with a fixed 3 bit priority level. The MAX uses these bits in determining priority levels when programmed for fixed priority mode of operation.

Arbitration always occurs on a clock edge, but only occurs on edges when a change in mastership will not violate AHB-Lite protocols. Valid arbitrations points include any clock cycle in which **sX_hready** is asserted (provide the master is not performing a burst or locked cycle) and any wait state in which the master owning the bus indicates a transfer type of IDLE (provided the master is not performing a locked cycle).

Since arbitration can occur on every clock cycle the slave port masks off all master requests if the current master is performing a locked transfer or a protected burst transfer, guaranteeing that no matter how low its priority level, it will be allowed to finish its locked or protected portion of a burst sequence.

16.4.4.4.3 Slave Port State Machine Master Handoff

The only times the slave port switches masters when programmed for fixed priority mode of operation is when a higher priority master makes a request or when the current master is the highest priority and it gives up the slave port by either running and IDLE cycle to the slave port or running a valid access to a location other than the slave port.

If the current master loses control of the slave port because a higher priority master takes it away, the slave port does not incur any wasted cycles. The current master has its current cycle terminated by the slave port at the same time as the new master's address and control information is recognized by the slave port. This looks like a seamless transition on the slave port.

If the current master is being wait stated when the higher priority master makes its request, the current master is allowed to make one more transaction on the slave bus before giving it up to the new master. Figure 16-8 illustrates the effect of a higher priority master taking control of the bus when the slave port is programmed for a fixed priority mode of operation.

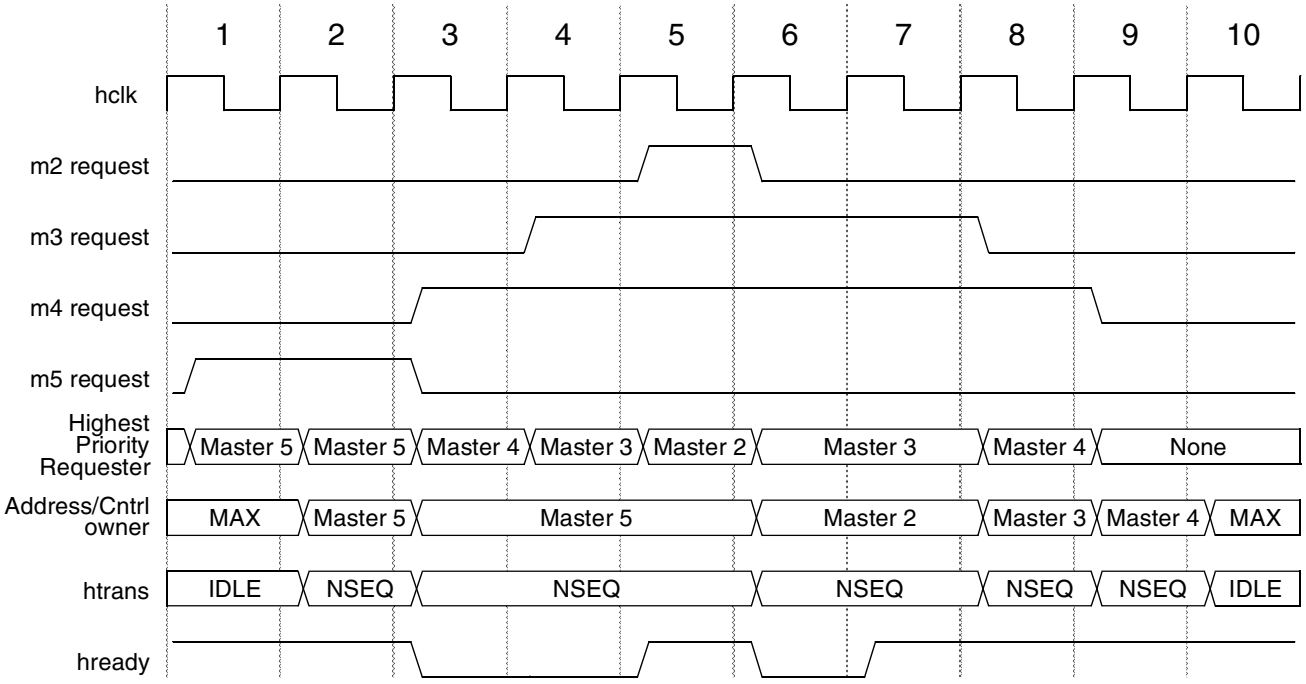


Figure 16-8. Low to High Priority Mastership Change

If the current master is the highest priority master and it gives up the slave port by running an IDLE cycle or by running a valid cycle to another location other than the slave port, the next highest priority master gains control of the slave port. If the current access incurs any wait states, the transition is seamless and no bandwidth will be lost. However, if the current transaction is terminated without wait states, one IDLE

cycle is forced onto the slave bus by the MAX before the new master is able to take control of the slave port. If no other master is requesting the bus, IDLE cycles are run by the MAX. However, no bandwidth is truly lost because no master is making a request.

Figure 16-9 illustrates the effect of a higher priority master giving up control of the bus.

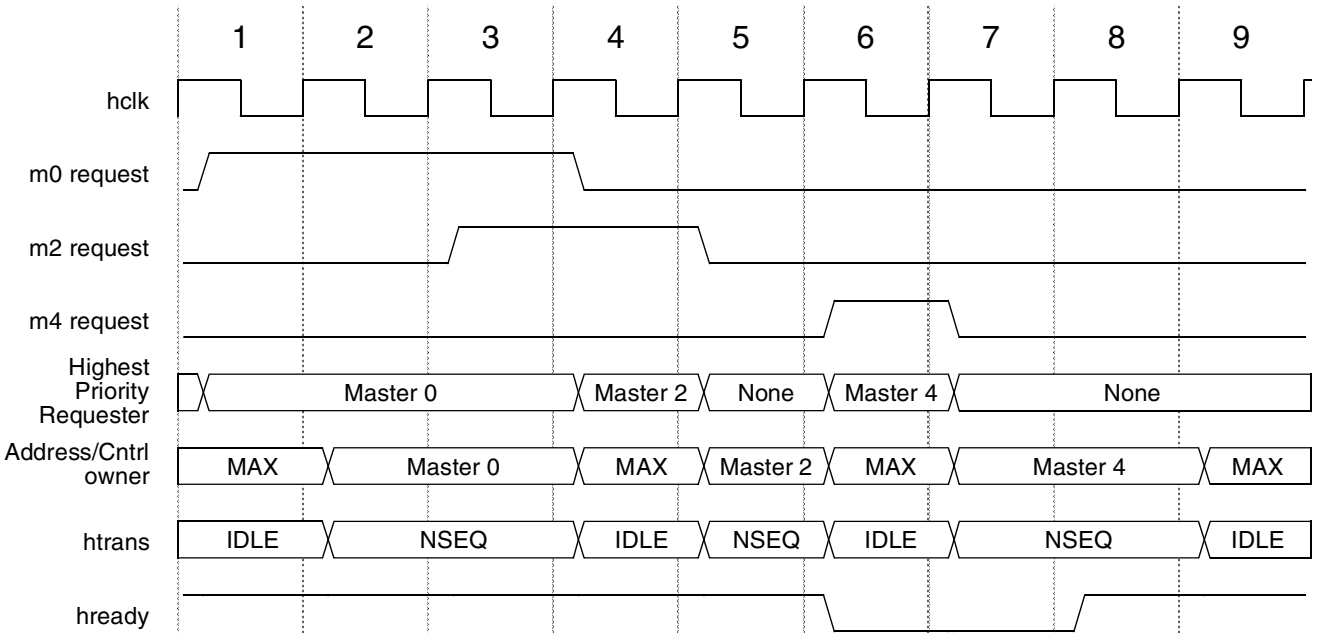


Figure 16-9. High to Low Priority Mastership Change

When the slave port is programmed for round-robin mode of arbitration, the slave port switches masters any time there is more than one master actively making a request to the slave port. This happens because

any master other than the one which presently owns the bus is considered to have higher priority. Figure 16-10 shows an example of round-robin mode of operation.

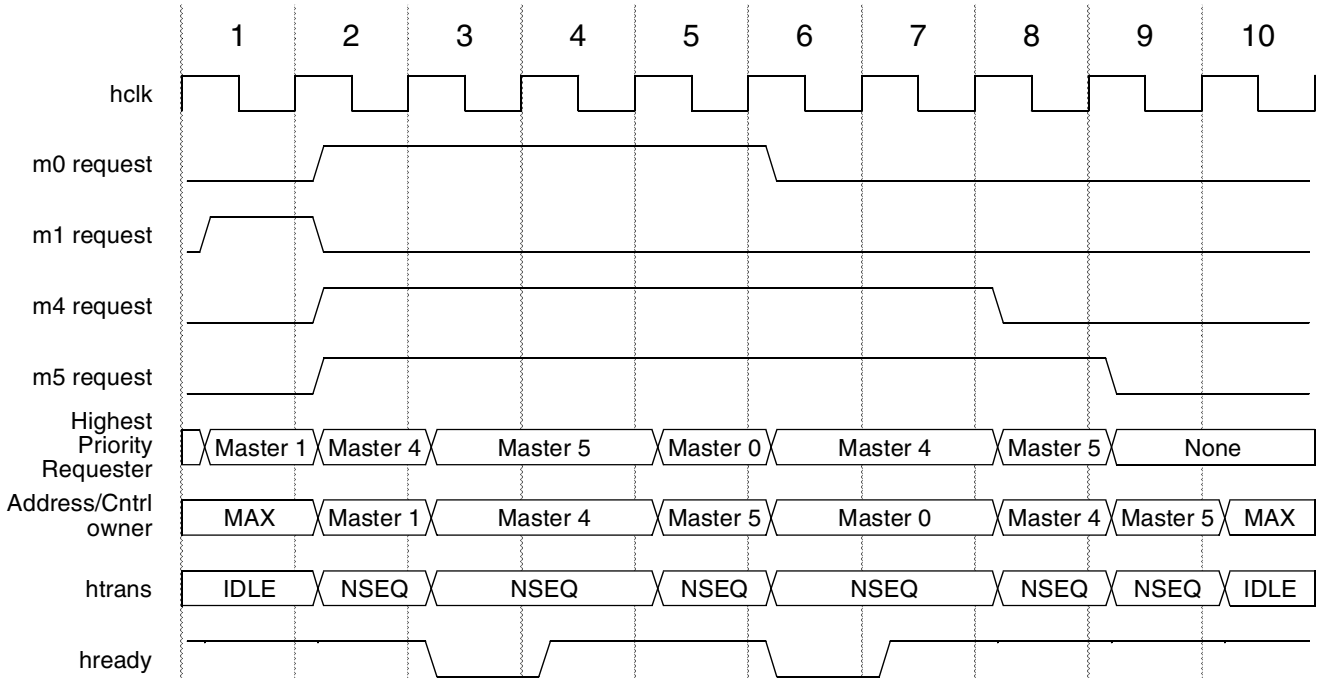


Figure 16-10. Round-Robin Mastership Change

16.4.4.4.4 Slave Port State Machine Parking

If no master is currently making a request to the slave port, the slave port parks in one of four places as dictated by the PCTL and PARK bits in the SGPCR and the locked state of the last master to access it.

If the last master to access the slave port ran a locked cycle and continues to run locked cycles even after leaving the slave port, the slave port parks on that master without regard to the bit settings in the SGPCR and without regard to pending requests from other masters. This is done so a master can run a locked transfer to the slave port, leave it, return to it, and be guaranteed that no other master has had access to it (provided the master maintains all transfers are locked transfers). If locking is not an issue for parking, the SGPCR bits dictate the parking method.

If the PCTL bits are set for low-power park mode, the slave port enters low-power park mode. It does not recognize any master as being in control of it, and it does not select any master's signals to pass through to the slave bus. In this case all slave bus activity effectively halts because all slave bus signals being driven from the MAX are 0. This of course can save quite a bit of power if the slave port is not in use for some time. The down side is that when a master does make a request to the slave port, it is delayed by one clock because it has to arbitrate to acquire ownership of the slave port.

If the PCTL bits are set to "park on last" mode, the slave port parks on the last master to access it, passing all that masters signals through to the slave bus. The MAX asynchronously forces **htrans[1:0]**, **hmaster[3:0]**, **hburst[2:0],** and **hmastlock** to 0 for all access that the master does not run to the slave port.

When that master accesses the slave port again, it does not pay any arbitration penalty; however, if any other master wishes to access the slave port, a one-clock arbitration penalty will be imposed.

If the PCTL bits are set to use PARK mode, the slave port parks on the master designated by the PARK bits. The behavior here is the same as for the park on last mode with the exception that a specific master is parked on instead of the last master to access the slave port. If the master designated by the PARK bits tries to access the slave port, it does not pay an arbitration penalty while any other master pays a one-clock penalty. Figure 16-11 illustrates parking on a specific master.

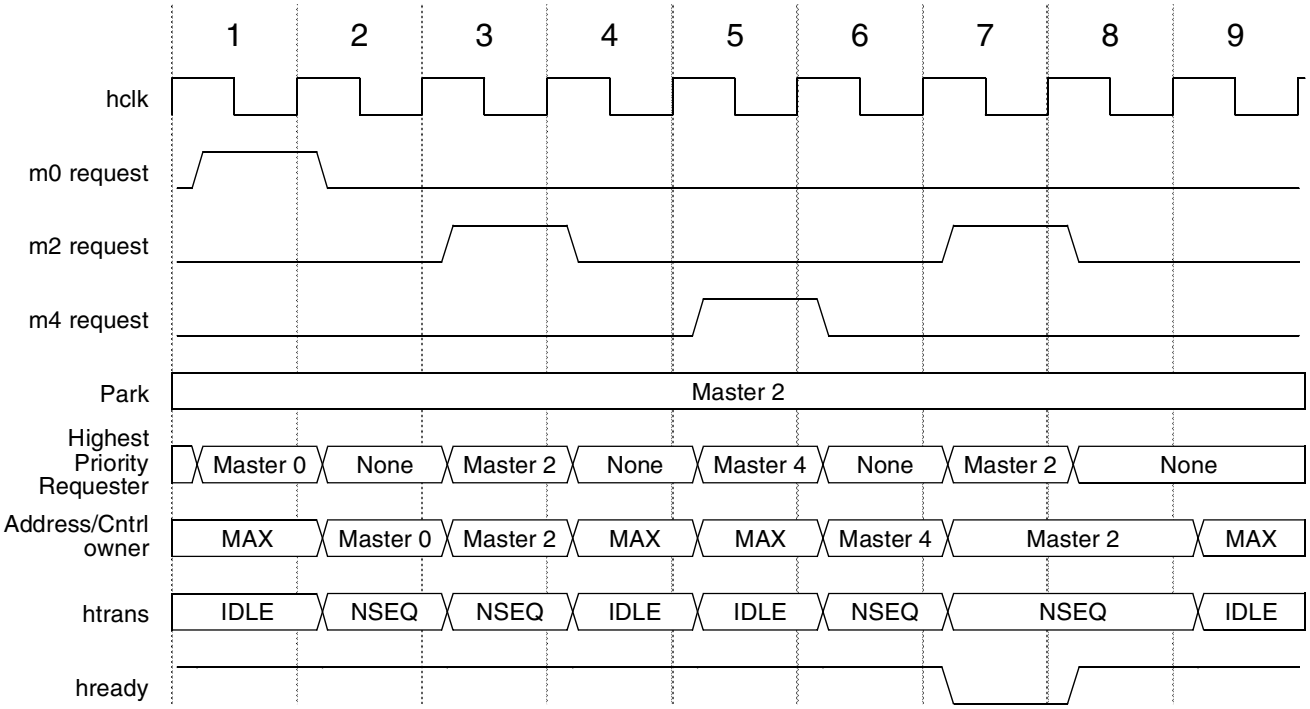


Figure 16-11. Parking on a Specific Master

Figure 16-12 illustrates parking on the last master. Note that in cycle 6 simultaneous requests are made by master 2 and master 4. Although master 2 has higher priority, the slave bus is parked on master 4 so master

4’s access is taken first. The slave port parks on master 2 once it has given control to master 2. This same situation can occur when parking on a specific master as well.

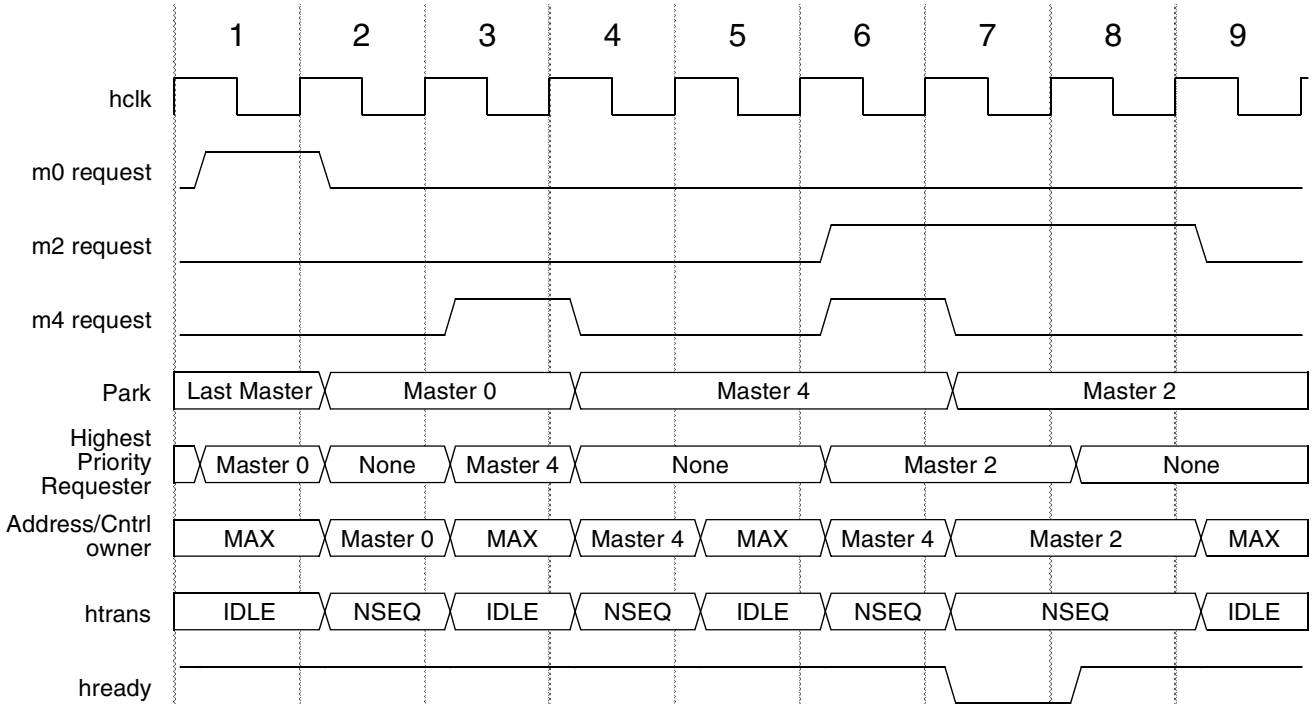


Figure 16-12. Parking on Last Master

16.4.4.4.5 Slave Port State Machine Halt Mode

If the **max_halt_request** input is asserted, the slave port eventually halts all slave bus activity and go into halt mode, which is almost identical to low power park mode. The HLP bit in the SGPCR controls the priority level of the **max_halt_request** in the arbitration algorithm. If the HLP bit is cleared, the **max_halt_request** has the highest priority of any master and gains control of the slave port at the next arbitration point (most likely the next bus cycle, unless the current master is running a locked or fixed length burst transfer). If the HLP bit is set, the slave port waits until no masters are actively making requests before moving to halt mode.

Regardless of the state of the HLP bit, once the slave port has gone into halt mode as a result of **max_halt_request** being asserted, it remains in halt mode until **max_halt_request** is negated, regardless of the priority level of any masters that may make requests.

In halt mode no master is selected to own the slave port so all the outputs of the slave port are set to 0.

16.5 Initialization/Application Information

No initialization is required by or for the MAX. Hardware reset ensures all the register bits used by the MAX are properly initialized.

16.6 MAX Interface

This section provides information on the MAX interface.

16.6.1 Overview

The main goal of the MAX is to increase overall system performance by allowing multiple masters to communicate in parallel with multiple slaves. In order to maximize data throughput it is essential to keep arbitration delays to a minimum.

This section examines data throughput from the point of view of masters and slaves, detailing when the MAX stalls the masters or inserts bubbles on the slave side.

16.6.2 Master Ports

Master accesses receive one of four responses from the MAX. They are either terminated, taken, stalled, or responded to with an error.

16.6.2.1 Terminated Accesses

A master access is terminated if the transfer type is IDLE. The MAX terminates the access, so it is not allowed to pass through the MAX.

16.6.2.2 Taken Accesses

A master access is taken if the transfer type is non IDLE and the slave port to which the access decodes is either currently servicing the master or is parked on the master. In this case the MAX is completely transparent, the master’s access is immediately seen on the slave bus, and no arbitration delays are incurred.

16.6.2.3 Stalled Accesses

A master access is stalled if the transfer type is non IDLE and the access decodes to a slave port that is busy serving another master, parked on another master or is in low power park mode. The MAX indicates to the master that the address phase of the access has been taken but then queues the access to the appropriate slave port to enter into arbitration for access to that slave port.

If the slave port is currently parked on another master or is in low power park mode and no other master is requesting access to the slave port then only one clock of arbitration is incurred. If the slave port is currently serving another master of a lower priority and the master has a higher priority than all other requesting masters then the master gains control over the slave port as soon as the data phase of the current access is completed (burst and locked transfers excluded). If the slave port is currently servicing another master of a higher priority then the master gains control of the slave port once the other master releases control of the slave port if no other higher priority master is also waiting for the slave port.

16.6.2.4 Error Response Terminated Accesses

A master access is responded to with an error if the transfer type is non-IDLE and the access decodes to a location not occupied by a slave port. Please refer to Section 16.6.3, Slave Ports," for information on locations that are not occupied by a slave port.

16.6.3 Slave Ports

The goal of the MAX with respect to the slave ports is to keep them 100% saturated when masters are actively making requests. In order to do this the MAX must not insert any bubbles onto the slave bus unless absolutely necessary.

There is only one instance when the MAX forces a bubble onto the slave bus when a master is actively making a request. This occurs when a higher priority master has control of the slave port and is running single clock (zero wait state) accesses while a lower priority master is stalled waiting for control of the slave port. When the higher priority master either leaves the slave port or runs an IDLE cycle to the slave port the MAX takes control of the slave bus and run a single IDLE cycle before giving the slave port to the lower priority master that was waiting for control of the slave port.

The only other times the MAX has control of the slave port is when the MAX is halting or when no masters are making access requests to the slave port and the MAX is forced to either park the slave port on a specific master or put the slave port into low power park mode.

In most instances when the MAX has control of the slave port, it indicates IDLE for the transfer type, negates all control signals, and indicates ownership of the slave bus via the **hmaster** encoding of 4'b0000. One exception to this rule is when a master running locked cycles has left the slave port but continues to run locked cycles. In this case the MAX controls the slave port and indicate sIDLE for the transfer type but it does not affect any other signals.

NOTE

When a master runs a locked cycle through the MAX, the master is guaranteed ownership of all slave ports it accesses while running locked cycles for one cycle beyond when the master finishes running locked cycles.

Chapter 17
Digital Audio Mux (AUDMUX)

The Digital Audio Mux (AUDMUX) provides a programmable interconnect device for voice, audio, and synchronous data routing between host serial interfaces (that is, SSI) and peripheral serial interfaces (that is, audio and voice codecs, also known as coder-decoders).

The AUDMUX allows the audio system connectivity to be modified through programming (as opposed to altering the PCB schematics of the system). Figure 17-1 and Figure 17-2 show the block diagram of the AUDMUX (Figure 17-2 is an extension of Figure 17-1).

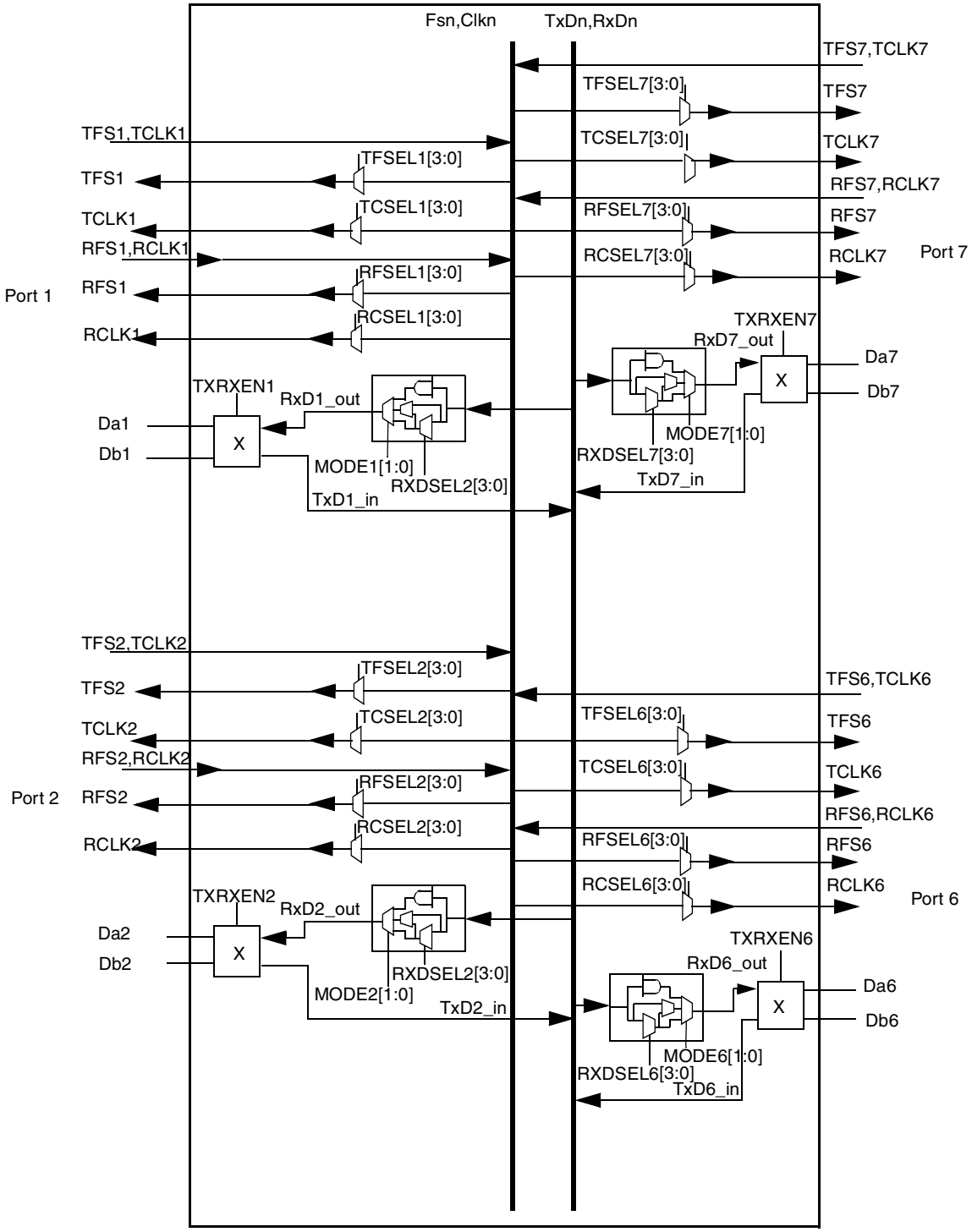


Figure 17-1. AUDMUX Block Diagram A

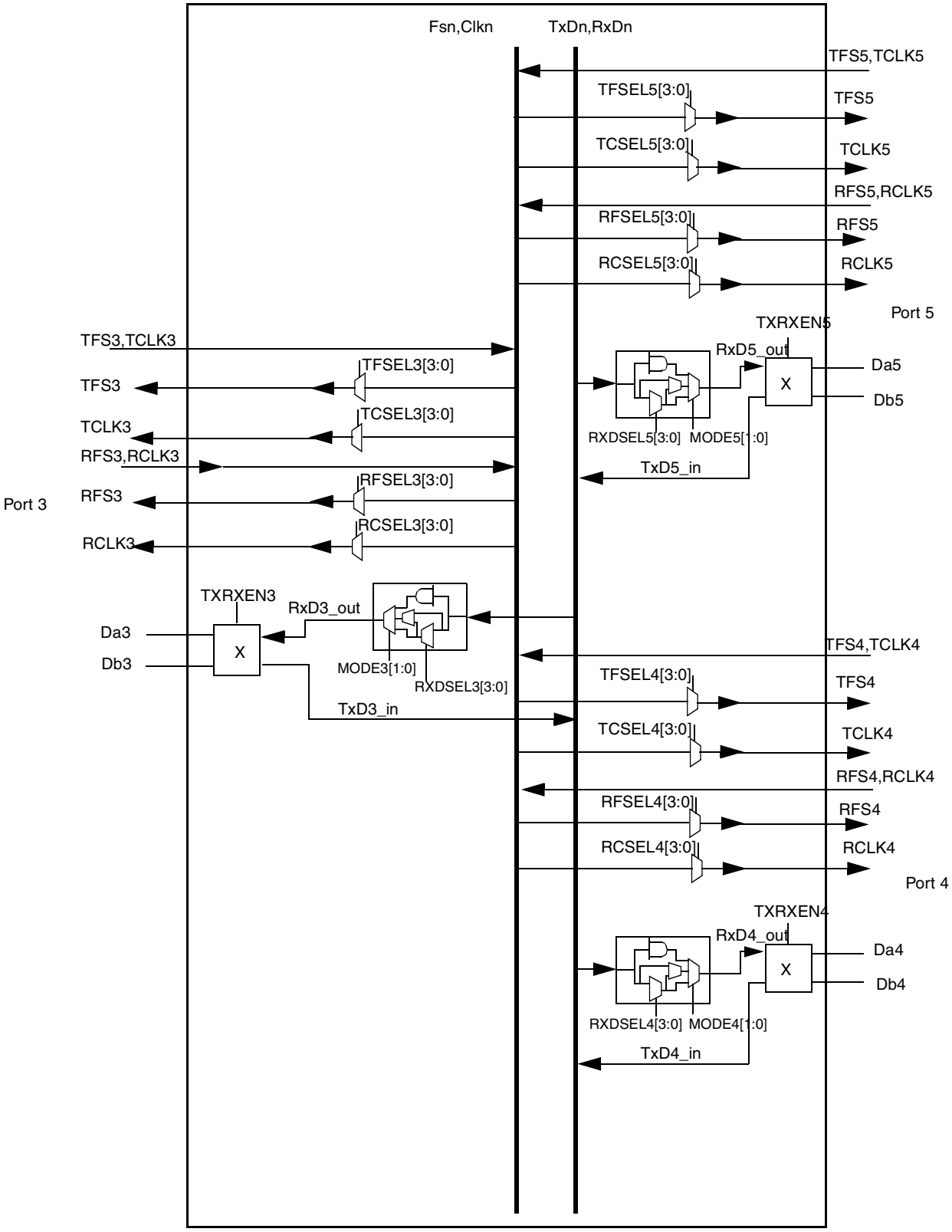


Figure 17-2. AUDMUX Block Diagram B

17.1 Introduction

With the AUDMUX, resources do not need to be hard-wired and can be effectively shared in different configurations. The AUDMUX interconnections allow multiple, simultaneous audio/voice/data flows between the ports in point-to-point or point-to-multipoint configurations.

The AUDMUX includes two types of interfaces. Internal ports connect to the processor serial interfaces and external ports connect to off-chip audio devices and serial interfaces of other processors. A desired connectivity is achieved by configuring the appropriate internal and external ports.

17.1.1 Features

- Four external ports
- Full 6-wire SSI interfaces for asynchronous receive and transmit
- Configurable 4-wire (synchronous) or 6-wire (asynchronous) peripheral interfaces
- Independent Tx/Rx Frame sync and clock direction selection for host or peripheral
- Each host interface's capability to connect to any other host or peripheral interface in a point-to-point or point-to-multipoint (network mode)
- Transmit and Receive Data switching to support external network mode

17.1.2 Modes of Operation

Figure 17-1 and Figure 17-2 show the AUDMUX block diagram.

All the ports are basically identical. The major difference is whether a port is connected to an on-chip serial interface (for example, SSI) or connected to the chip's pads to connect to off-chip serial devices (that is, any 4-wire or 6-wire external SSI, voice, I2S, or AC97 CODEC).

Port 7 can be physically connected to any of the peripherals previously mentioned. There is no functional difference among Ports 1 through 7.

All ports can be configured as four- or six-wire interfaces. When configured as a six-wire interface, the additional RFS and RCLK signals of the interface enable the serial interface to be used in asynchronous mode with separate receive and transmit clocks.

All ports have a Tx/Rx switch to provide flexibility in supporting network mode configurations. The Tx/Rx switch enables the transmit and receive data lines to be swapped so that mastership of the serial bus can be passed among multiple external devices connected to a single port.

All ports, in addition to supporting the default external network mode, support —internal network mode. With internal network mode, a point-to-multipoint network configuration with an arbitrary number of slaves can be supported if the external slaves are put into the high-impedance state (as defined in the SSI network mode protocol) and have pull-up resistors on their TxD pins. (Alternatively, this can be viewed as requiring a pull-up resistor on the corresponding AUDMUX RxD pin.)

Bit clock direction selection enables each port to be configured as a master or slave in the flow.

Possible scenarios include:

- SSI1 (internal port) drives a voice CODEC and a BT CODEC (both on external Port 6) and the Bottom Connector (on external Port 7) simultaneously using network mode. SSI1 is the master.
- An external processor (external port - Port 5) drives a voice CODEC and a BT CODEC (both on external Port 6) and the Bottom Connector (on Port 7) simultaneously using network mode. SAP is the master.

NOTE

SSI1 (internal port), which is the master, drives a voice CODEC and a BT CODEC (both on external Port 6) and the bottom connector (on external Port 7) simultaneously using network mode.

An external processor (external port - Port 5), which is the master, drives a voice CODEC and a BT CODEC (both on external Port 6) and the bottom connector (on Port 7) simultaneously using network mode.

17.1.2.1 Port Receive Data Modes

Each port has logic to select which data lines are used to create the RxD line for the corresponding host interface. Figure 17-3 shows the logic used to create the RxD line for Port 1. This logic has the following modes of operation (as determined by MODE0):

- Normal
- Internal network mode

The subsequent sections describe the various modes of the port receive data logic. The following terms are used to define the operation of the AUDMUX:

Network mode— Time-Division Multiplexed protocol for sending unique data to multiple devices on a serial bus.

Internal network mode—Physical bus configuration where multiple serial buses are effectively connected within the AUDMUX via digital logic to create point-to-multipoint connectivity. An arbitrary number of devices are supported. Devices must be put into the high-impedance state as specified by the network mode protocol. TxDATA lines of devices must be pulled high.

External network mode—Physical bus configuration where multiple serial buses are electrically connected together on a printed circuit board (that is, external to the AUDMUX). Devices must put their TxDATA lines into the high-impedance state as specified by the network mode protocol.

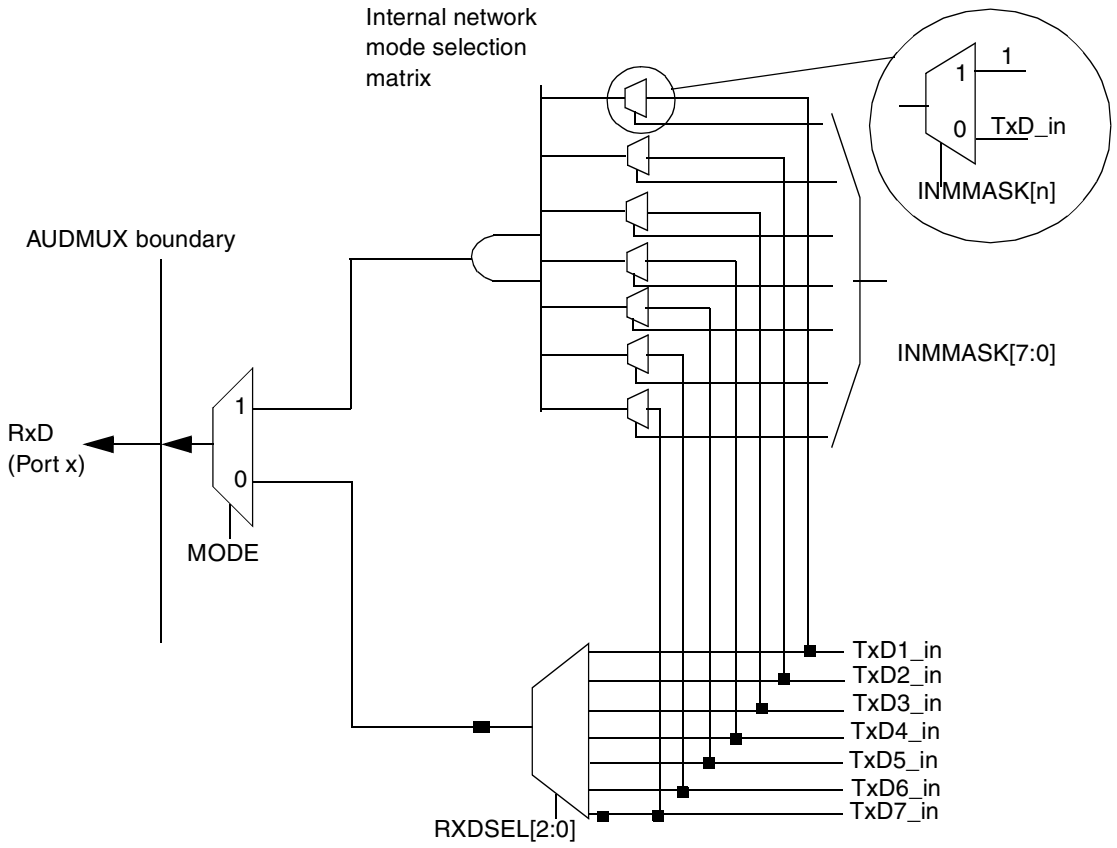


Figure 17-3. Receive Data Logic for Port x

17.1.2.1.1 Normal Mode

In normal mode (MODE0=0), the port is connected in a point-to-point configuration (as a master or a slave) and the RXDSEL[2:0] setting selects the transmit signal from any port. In normal mode, any data format can be used (that is, SSI normal mode, SSI network mode, AC-97, and others.

17.1.2.1.2 Internal Network Mode

In internal network mode (MODE0 = 1), the output of the AND gate is routed (via the output of the port) to the RxD signal of the corresponding host interface. The INMMASK bit vector selects the transmit signals of the ports that are to be connected in network mode. The transmit signals received at the AUDMUX ports (TxDn_in) are ANDed together to form the output. In internal network mode, only one device can be transmitting in its predesignated timeslot and all other transmit signals must remain high (be in high-impedance state and pulled-up). Therefore, non-active signals in the selection will be high and do not influence the output of the AND gate.

Network mode is a protocol where a master SSI is connected to more than one slave SSI device and communication occurs on a time-slotted frame. Though network mode can allow master-slave and slave-slave communication, internal network mode supports only master-slave communication.

There are two scenarios where internal network mode can be used with external network mode:

1. Slave-only devices are attached to an external port.
2. A master device is attached to an external port and all slave devices connected to the same external port are disabled.

NOTE

When internal network mode is enabled at an external port, RXDSEL[3:0] for RxDn_obe selection is ignored and RxD_obe is always driven high (that is, asserted for all timeslots). All slave devices connected to the same port must be disabled.

Internal Network Mode Example 1

SSI1 and SSI2 are used with Port 4 in internal network mode as shown in Figure 17-4. No pull-up resistors are required, since all the interfaces combined in internal network mode are on-chip interfaces. The on-chip interfaces drive a logic ‘1’ when their output enables are logic ‘0’.

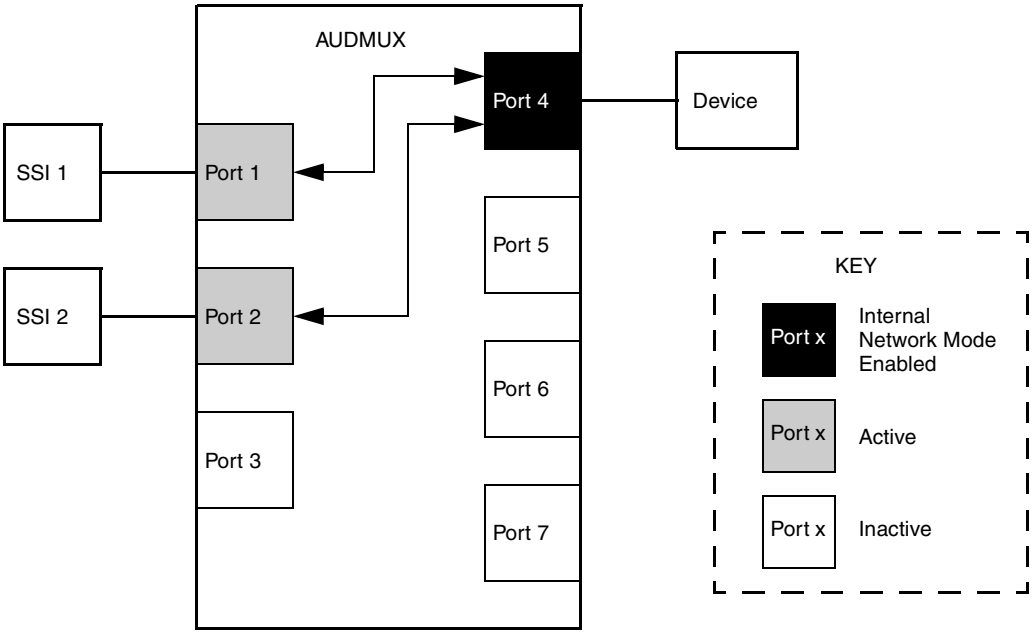


Figure 17-4. Block Diagram For Example 1

See Figure 17-5 for the timing diagram of Example 1. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

The data lines for SSI1 and SSI2 (as well as their output enables) are shown. Note that the SSI transmits a logic ‘1’ when its corresponding output enable is a logic ‘0’. The combined TxDATA line, which is the logical AND of SSI1 and SSI2’s TxDATA lines, is used for Port 4’s TxDATA line.

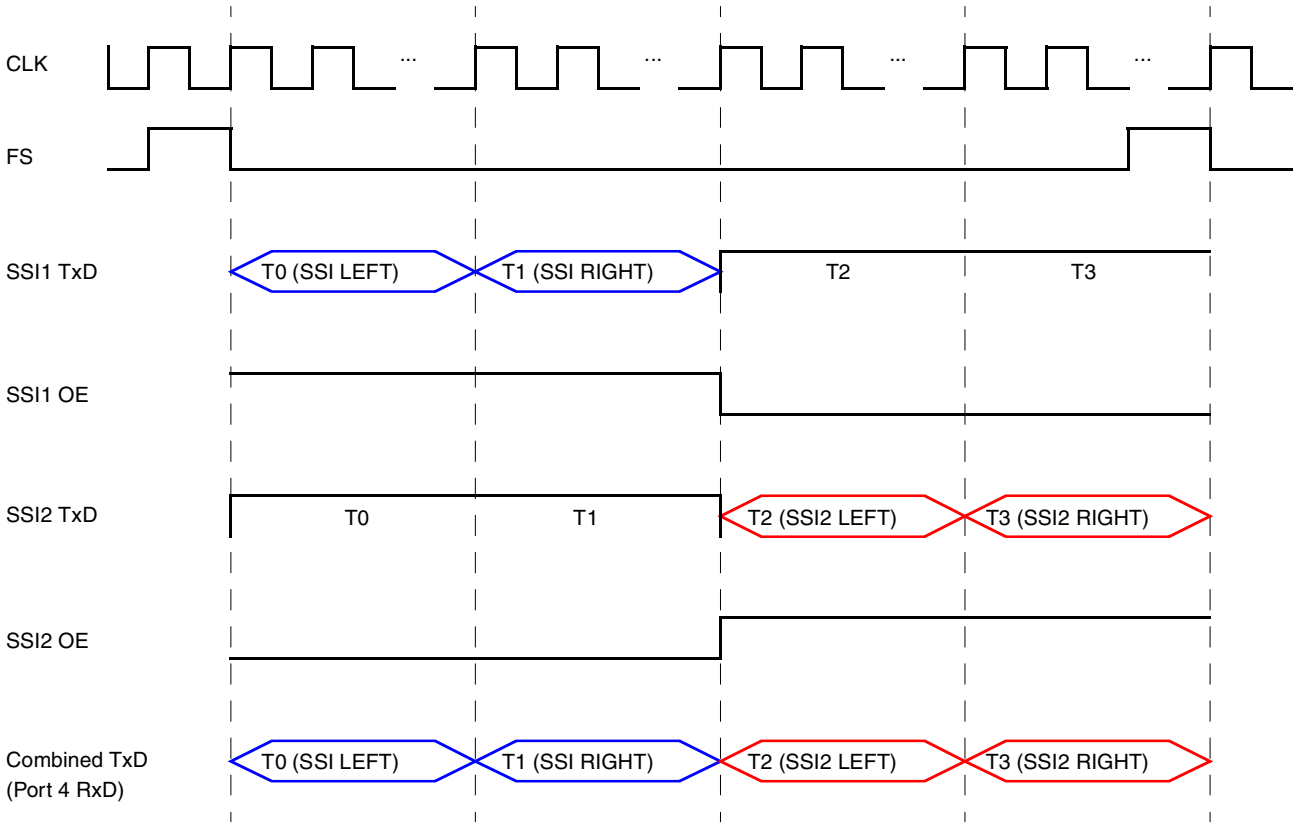


Figure 17-5. Example Using All Internal Ports For Transmit Data

Internal Network Mode Example 2

The SSI, Port 4, and Port 5 are used with Port 6 in internal network mode, as shown in Figure 17-6. Note that Port 4 and Port 5 are external ports. Therefore, pull-up resistors are required on the Port 4 RxDATA and Port 5 RxDATA pins. This example shows the timing associated with using adjacent timeslots for the SSI, Port 4, and Port 5.

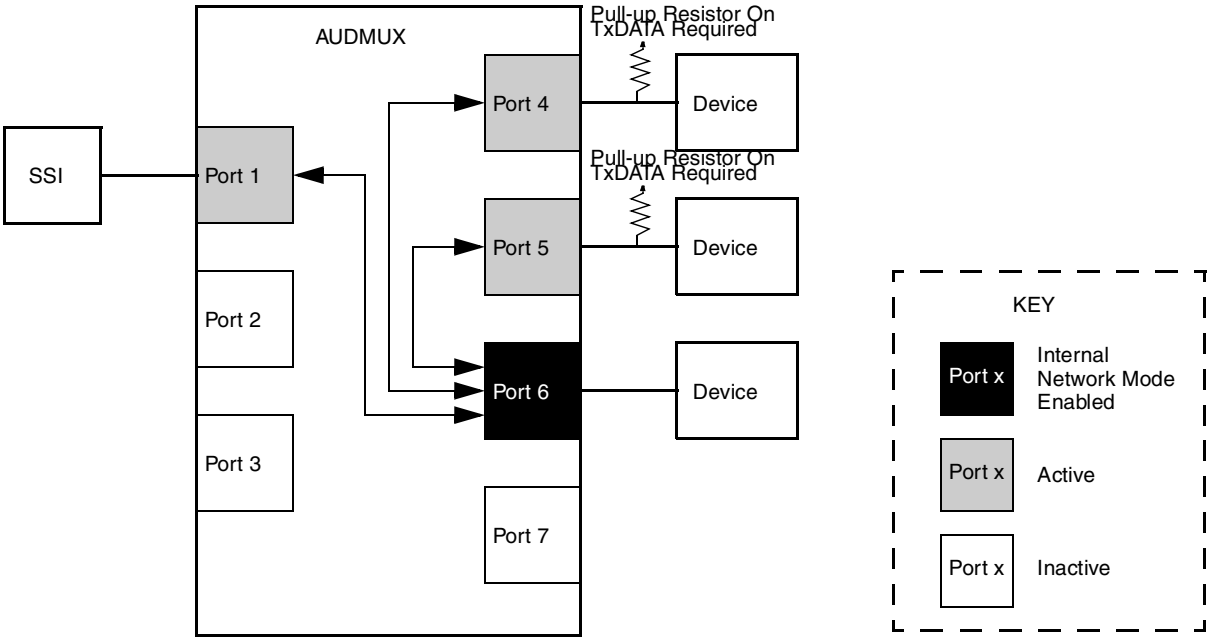


Figure 17-6. Block Diagram For Example 2

The resistance value of the pull-up resistors must be sufficiently high such that a value of ‘0’ can be pulled up to logic ‘1’ within half of a period of the bitclock. The required resistance must be no larger than:

Rmax = $1 \div (2 \times f_{bc} \times C)$ where:

f_{bc} is the frequency of the bitclock

C is the total system capacitance (ICs, board traces, and so on)

Figure 17-7 shows the timing diagram for this example. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

The data lines for the SSI, Port 4, and Port 5 are shown. Note that the SSI transmits a logic ‘1’ when its corresponding output enable is a logic ‘0’. The data lines from Port 4 and Port 5 *at the pad* are pulled high by pull-up resistors when they are in the high-impedance state. The data lines from Port 4 and Port 5 *at the AUDMUX* are pure digital signals and are constantly driven. The combined TxDATA line, which is the logical AND of the SSI, Port 4, and Port 5’s TxDATA lines, is used for Port 6’s TxDATA line.

Note the highlighted areas in Figure 17-7. This shows the transition time that occurs while a TxDATA line is being pulled high. In this example, this transition time is a maximum of 1/2 the period of the serial bitclock. This prevents corruption of the first data bit of the next timeslot. It is critical that the pull-up resistance is sufficient for the given bitclock frequency and system capacitance.

Note that hysteresis should be enabled at Port 4’s RxDATA pad and Port 5’s RxDATA pad to prevent the digital signals created by the pad from toggling rapidly during the pull-up period. The pads typically require a transition within 25 ns unless hysteresis is enabled. Instead of using hysteresis, one could select a pull-up resistor sufficiently high to pull-up the signal at the pad within 25 ns; however, that would result in a higher resistance value and higher current drain.

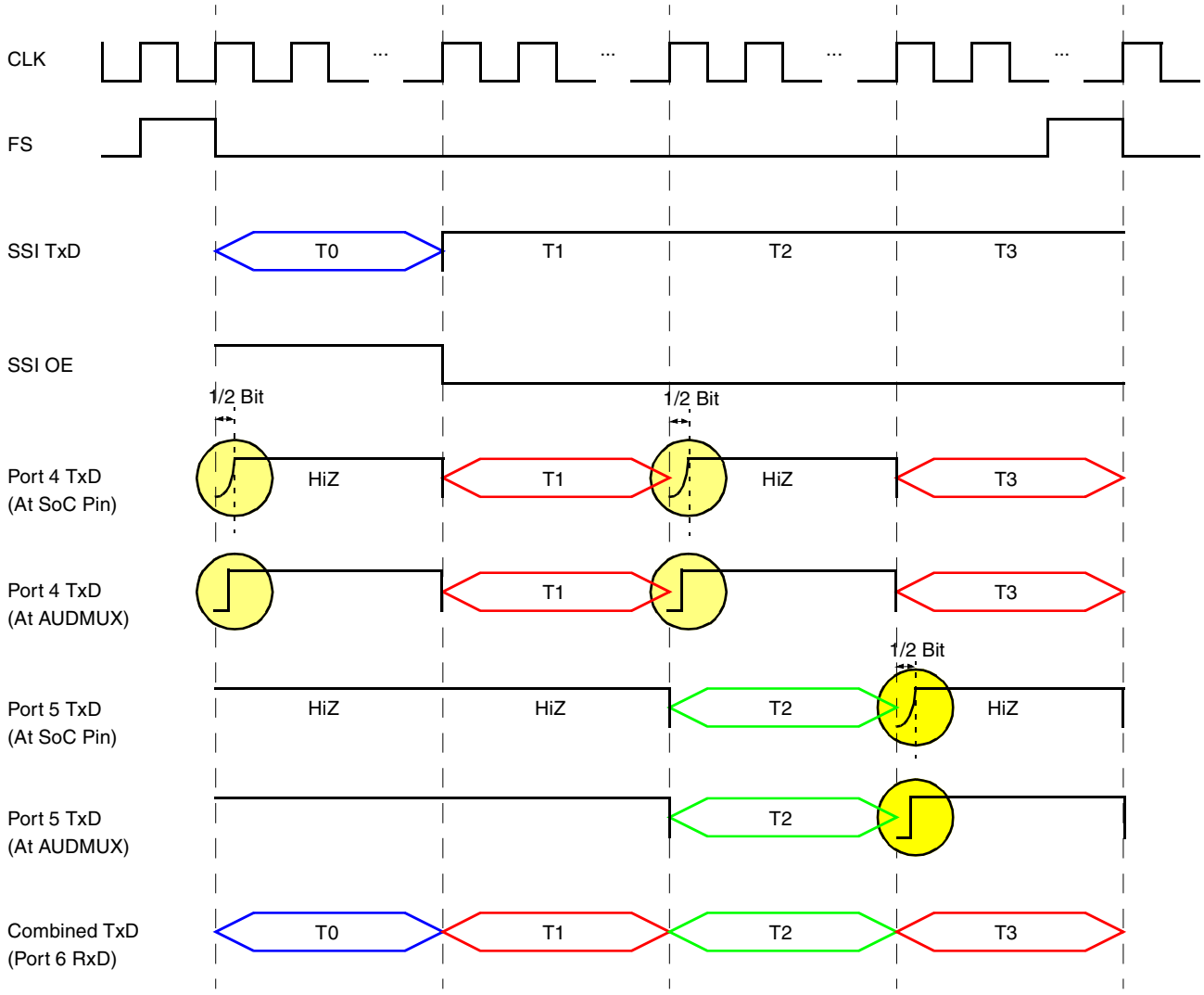


Figure 17-7. Example Using External Ports for Transmit Data in Consecutive Timeslots

Internal Network Mode Example 3

The SSI and Port 4 are used with Port 6 in internal network mode as shown in Figure 17-8. Note that Port 4 is an external port. Therefore, a pull-up resistor is required on the Port 4 TxDATA pin. This example shows the timing associated with inserting empty timeslots after the timeslots have been used by external ports.

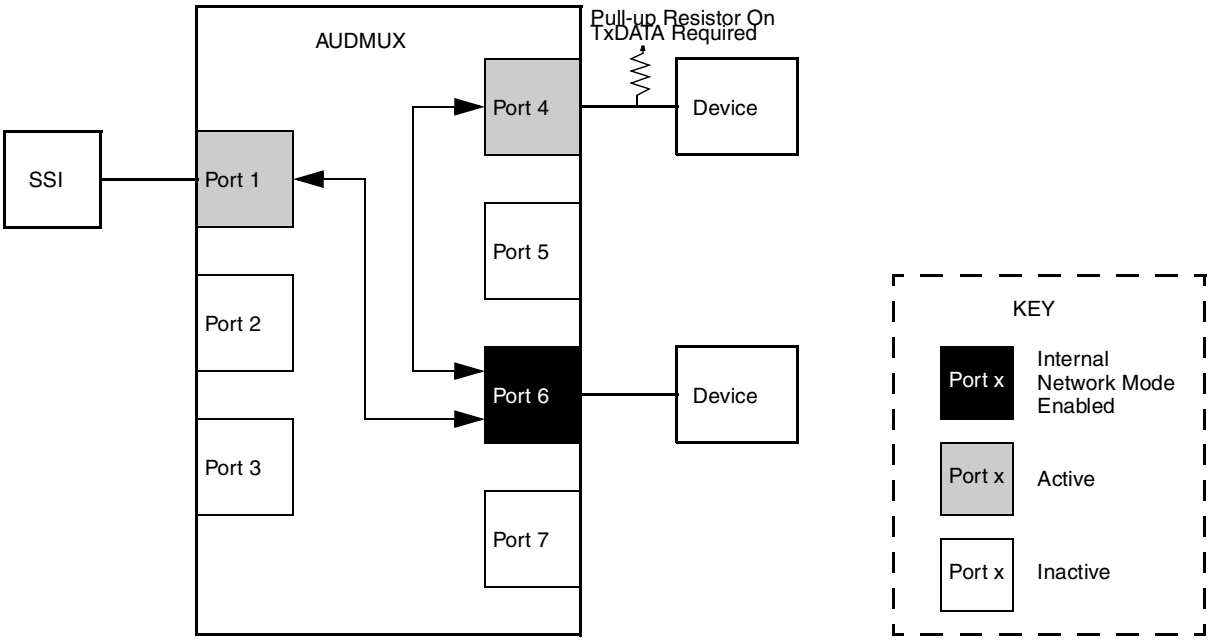


Figure 17-8. Block Diagram For Example 3

The resistance value of the pull-up resistors must be sufficiently high such that a value of ‘0’ can be pulled up to logic ‘1’ by the time that the next occupied timeslot occurs. This allows a much weaker pull-up to be used as compared to Example 2. The required resistance must be no larger than:

$Rmax = (4 \times n + 1) \div (2 \times f_{bc} \times C)$ where:

n is the number of bits per timeslot

f_{bc} is the frequency of the bitclock

C is the total system capacitance (ICs, board traces, and so on)

Figure 17-9 shows the timing diagram for this example. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

The data lines for the SSI and Port 4 are shown. Note that the SSI transmits a logic ‘1’ when its corresponding output enable is a logic ‘0’. The data line from Port 4 *at the pad* is pulled high by a pull-up resistor when they are in the high-impedance state. The data line from Port 4 *at the AUDMUX* is a pure digital signal and is constantly driven. The combined TxDATA line, which is the logical AND of the SSI and Port 4’s TxDATA lines, is used for Port 6’s RxDATA line.

Note the highlighted area in Figure 17-9. This shows the transition time that occurs while Port 4’s TxDATA line is being pulled high. In this example, this transition time is a maximum of two timeslots plus 1/2 the period of the serial bitclock. This prevents corruption of the first data bit of the next timeslot. It is critical that the pull-up resistance is sufficient for the given bitclock frequency and system capacitance.

Note that hysteresis must be enabled at Port 4’s RxDATA pad to prevent the digital signal created by the pad from toggling rapidly during the extended pull-up period. The pads typically require a transition within 25 ns unless hysteresis is enabled.

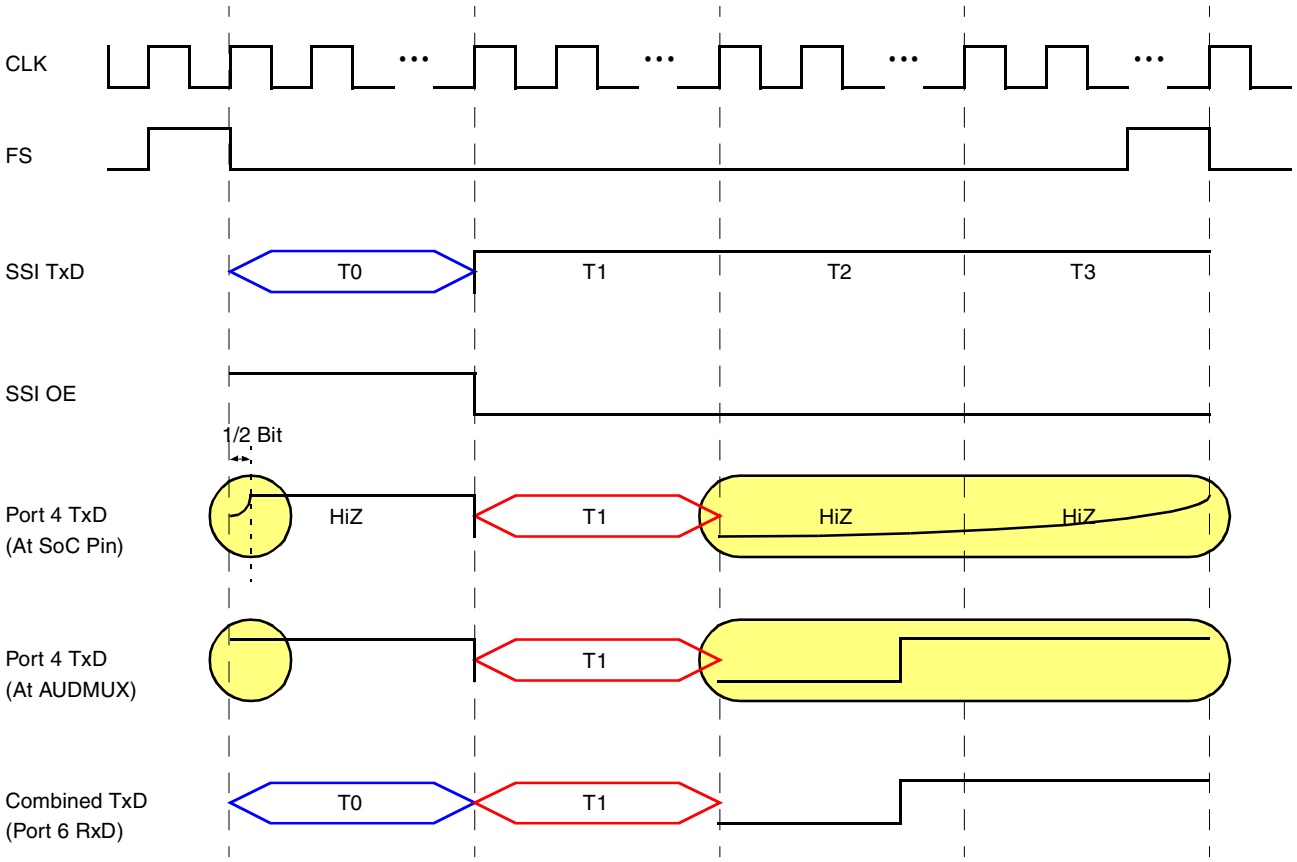


Figure 17-9. Example Using External Ports For Transmit Data In Nonconsecutive Timeslots

17.1.2.1.3 Transmit Data Output Enable Assertion

The TxDATA line from the internal network mode master (connected at any internal port) is put into the high-impedance state at the pad depending upon the assertion or deassertion of TxD_obe, its corresponding output enable generated by the network mode master.

In the case of an external network mode master (connected at an external port), the corresponding TxD_obe is always asserted after the port data register configuration.

17.1.2.2 Tx/Rx Switch and External Network Mode

External network mode is the traditional network mode connection. It is called external network mode to differentiate from the internal network mode. In external network mode, devices are connected to a single external port in a star or multi-drop configuration.

In network mode, there can be only one master (driving the frame sync and clock source) with the other devices configured in normal slave mode or network slave mode. Unlike internal network mode, both master-slave and slave-slave communication can take place in external network mode. CODEC devices transmit on a single timeslot while processor serial interfaces (that is, SSI, SAP) can process more than one timeslot of data while in network master or slave mode.

Figure 17-10 shows the Tx/Rx data switch. RxD_obe is the output buffer enable signal and RxD_out is the data transmit signal from the serial interface. The TxD_in signal is the receive data signal going towards the RXDSEL muxes of all ports.

D_TxRx is the data pin which serves as the chip-level transmit data pin when the TxRx switch is not enabled. D_RxTx is the data pin which serves as the chip-level receive data pin when the TxRx switch is not enabled. The roles of these pins are reversed when the TxRx switch is enabled.

When TXRXEN is disabled (TXRXEN = 0), RxD_out is routed to D_TxRx and D_RxTx is routed to TxD_in. The output buffer enable, selected by RXDSEL[2:0], is routed to Db_obe.

When the Tx/Rx switch is enabled (TXRXEN = 1), RxD_out is routed to D_RxTx and D_TxRx is routed to TxD_in. The output buffer enable, selected by RXDSEL[2:0], is routed to Da_obe.

If the RXDSELn[2:0] field for any Port *n* is configured to select data from an internal port, the output buffer enable is selected by RXDSELn[2:0] and is routed to Dan_obe/Dbn_obe. In the case when the RXDSELn[2:0] field for Port *n* is configured to select data from an external port, the output buffer enable is always high and routed to Dan_obe/Dbn_obe, depending on the TXRXENn switch configuration.

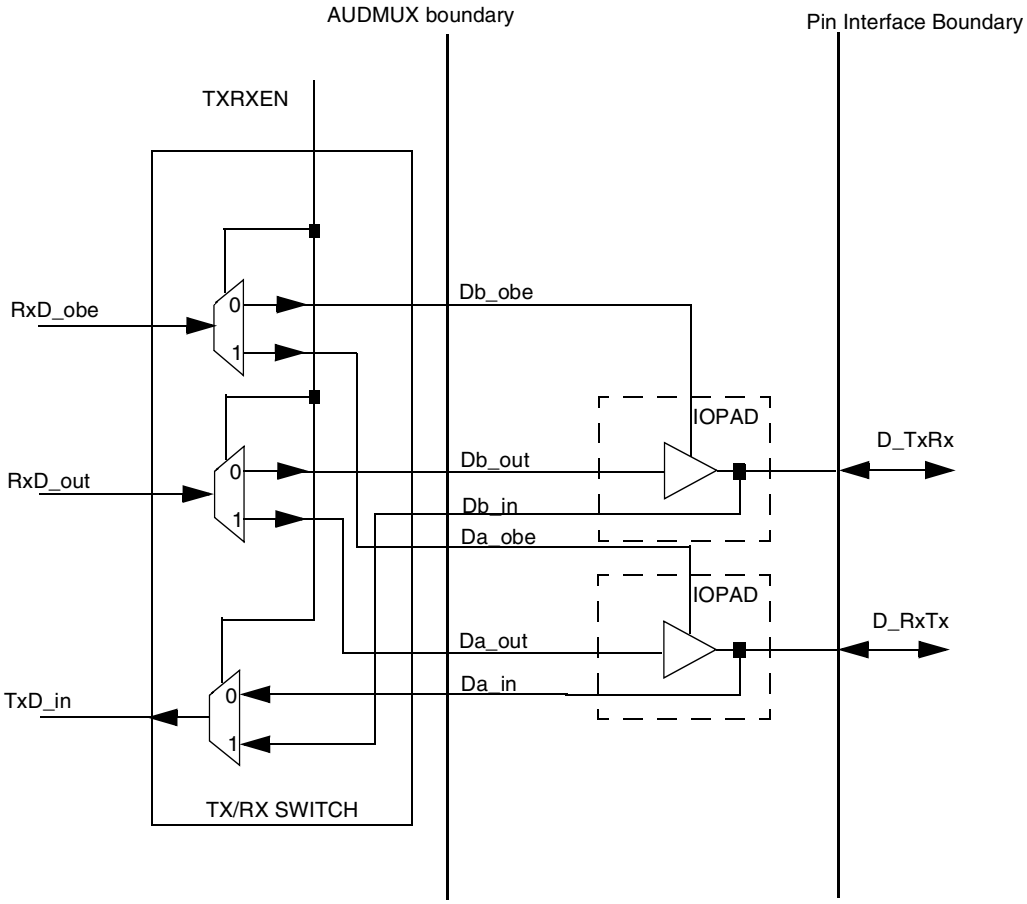


Figure 17-10. Tx/Rx Switch

17.1.2.3 Timing Modes

The AUDMUX ports are constructed as 6-wire interfaces. However, they can be used either in synchronous or asynchronous modes as determined by the SYN bit.

17.1.2.3.1 Synchronous Mode (4-wire Interface)

In Synchronous mode, the port has a 4-wire interface (that is, RxD, TxD, TxCLK, TxFS). The receive data timing is determined by TxCLK and TxFS.

As shown in Figure 17-11, Port x signals can be routed to Port y, producing 6-wire to 4-wire port connectivity.

TFS_in, RFS_in, TCLK_in, and RCLK_in are the input frame sync and bit clocks from the serial interface (Port x) with their corresponding output buffer enable signals (_obe). TFS_out, RFS_out, TCLK_out, and RCLK_out are the frame sync and bit clocks that are transmitted to the serial interface from the other ports.

The TFS_out and TCLK_out are selected at Port x by the TFSEL and TCSEL mux settings, respectively. RFS_out and RCLK_out are selected at Port x by the RFSEL and RCSEL mux settings, respectively. Similarly, in the external direction, Port y is configured as a 4-wire port; TFSEL selects the FS_obe and FS_out signals. In this mode, the configuration of RFSEL and RCSEL is not used, since the RFS_out and RCLK_out pins at Port y are not available.

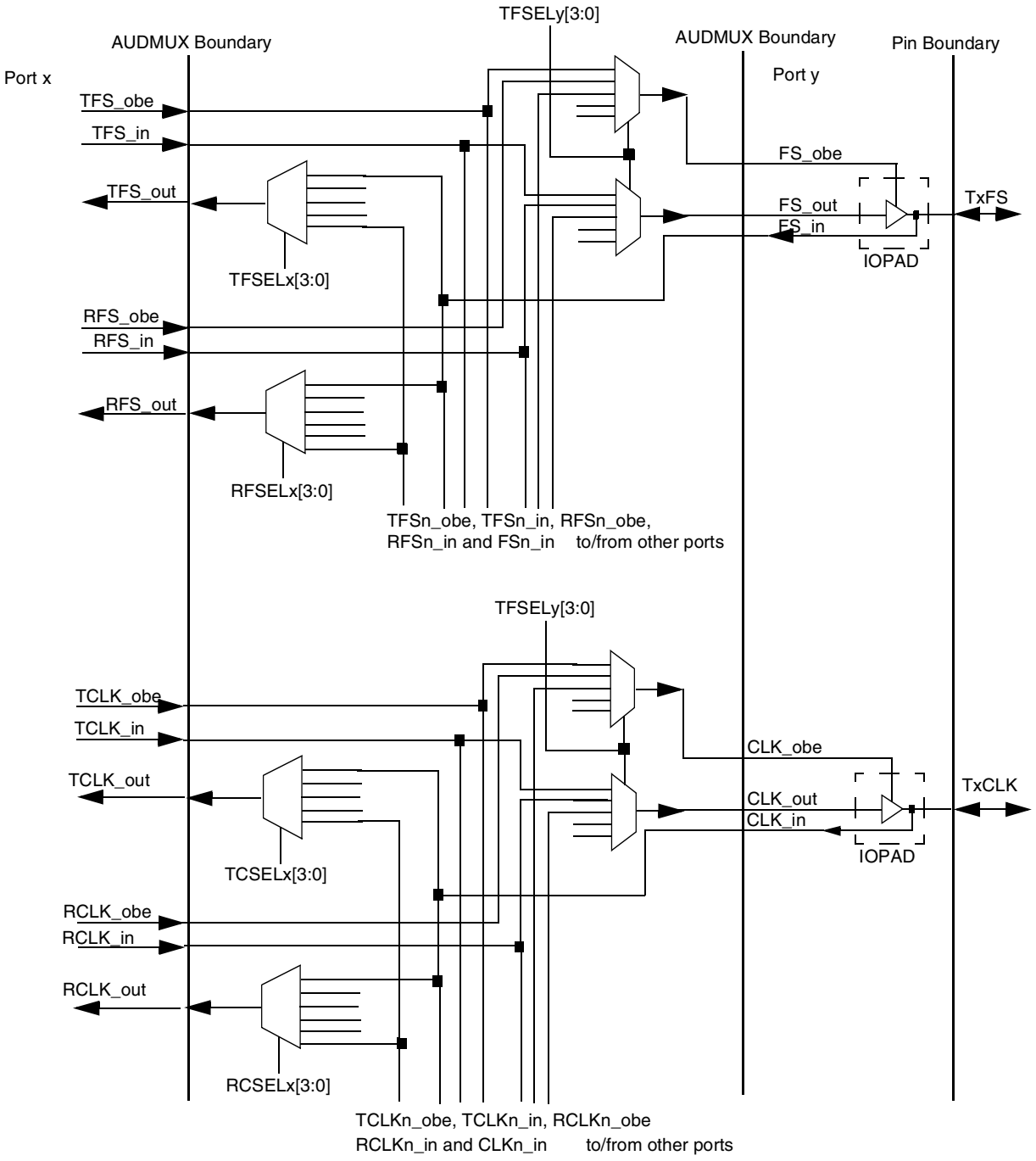


Figure 17-11. Frame Sync and Clock Routing When External Port Is 4-wire

17.1.2.3.2 Asynchronous Mode (6-wire Interface)

In Asynchronous mode, the port has a 6-wire interface (meaning RxD, TxD, TxCLK, TxFS, RxCLK, RxFS). This mode has additional receive clock (RxCLK) and frame sync (RxFS) signals as compared to the synchronous or 4-wire interface.

As shown in Figure 17-12 and Figure 17-13, Port x signals can be routed to Port y, producing 6-wire to 6-wire port connectivity.

TFS_in, RFS_in, TCLK_in, and RCLK_in are input frame sync and bit clocks from the serial interface (Port x) with their corresponding output buffer enable signals (_obe). TFS_out, RFS_out, TCLK_out, and RCLK_out are the frame sync and bit clocks that are transmitted to the serial interface from the other ports.

TFS_out and TCLK_out are selected by the TFSEL and TCSEL mux settings, respectively. RFS_out and RCLK_out are selected by the RFSEL and RCSEL mux settings, respectively. Similarly, in the external direction, the TFSEL selects the TxFS_obe and TxFS_out signals and TCSEL selects the TxCLK_obe and TxClk_out signals. The RFSEL selects the RxFS_obe and RxFS_out signals and RCSEL selects the RxCLK_obe and RxCLK_out signals.

NOTE

Since FS_in and CLK_in from external interfaces are also routed to the TFSEL and TCSEL muxes of the external ports, respectively, these signals do not have corresponding buffer enable signals. Consequently, their corresponding inputs to the TFSEL and TCSEL mux of the external ports have to be tied high.

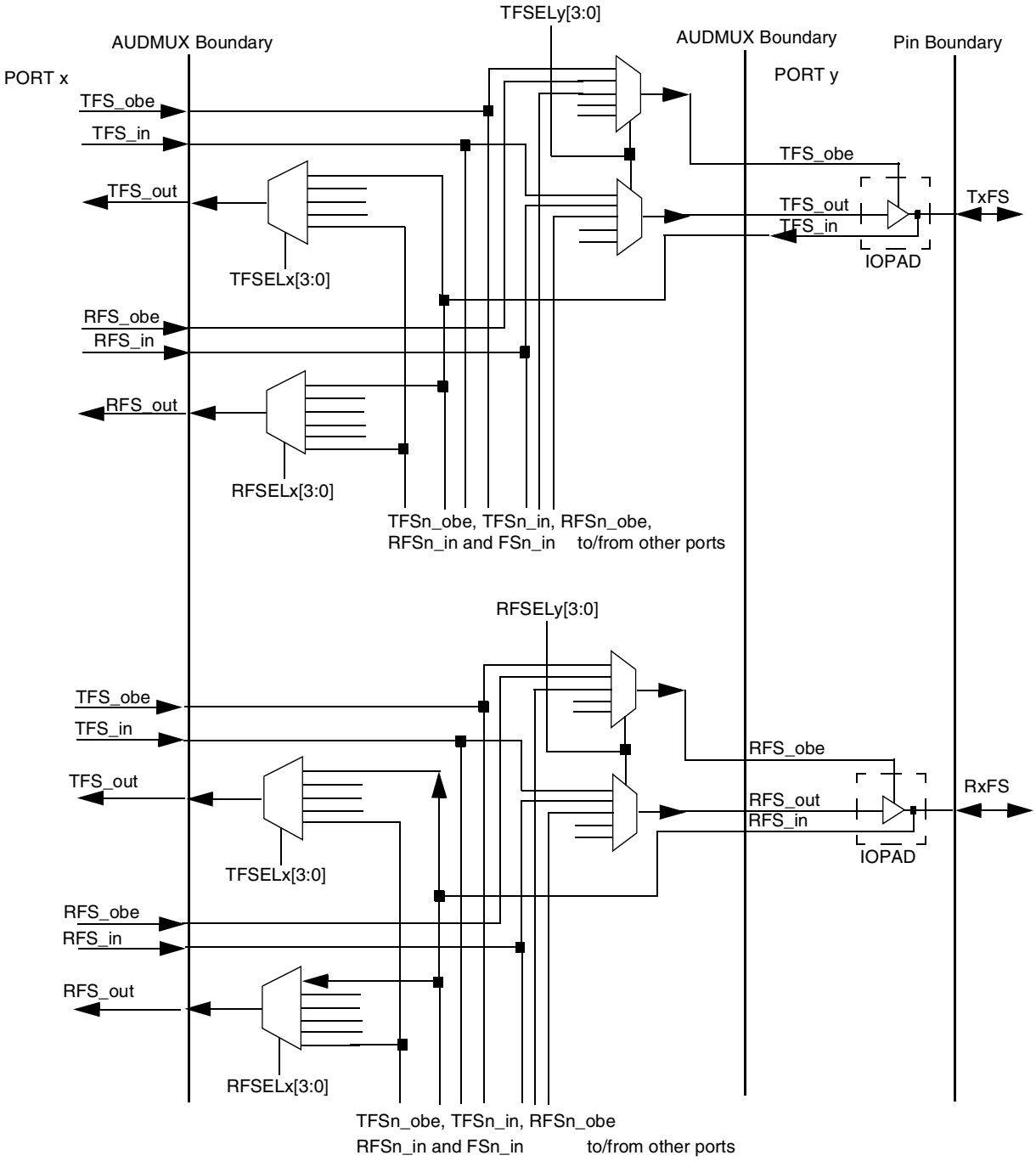


Figure 17-12. Frame Sync Routing When External Port Is 6-wire

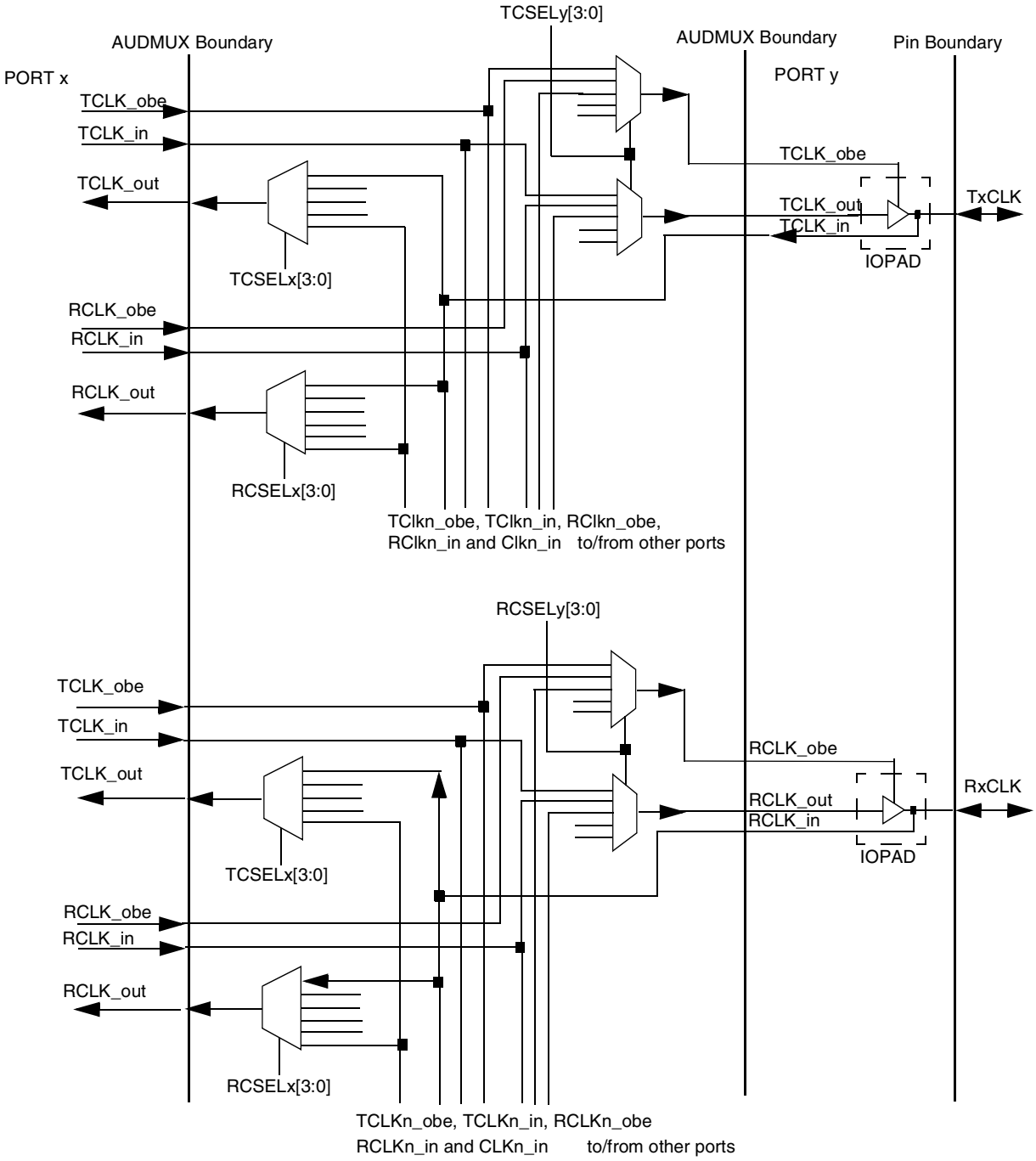


Figure 17-13. Clock Routing When External Port Is 6-wire

17.1.3 Connectivity Between Ports

Four basic types of connections are provided by the AUDMUX:

- Internal port to external port
- External port to external port

- Internal port to internal port
- Loopback

The corresponding data connections are described in the following sections.

17.1.3.1 Internal Port to External Port Connectivity

Figure 17-14 shows the data path connections between an internal port and an external port. The internal port is connected to a processor's serial interface. TxD_obe is the buffer enable signal from the serial interface, TxD_in is the input transmit data from the serial interface to the AUDMUX, and RxD_out is the receive data output from the AUDMUX to the serial interface.

RXDSEL[2:0] of the external port selects the buffer enable signal (TxD_obe) and transmit data output (TxD_out) signal from the TxD_obe and RxD_in signals. RXDSEL[2:0] is a common signal to both selection muxes.

NOTE

Since buffer TxD_in signals from external interfaces do not have corresponding buffer enable signals, their buffer enable signals into the selection mux are tied high. This will ensure that selection of TxD_in, as RxD_out will also drive the RxD_obe output high.

Transmit Data from the serial interface goes into the RXDSEL data mux and comes out as RxD_out. RxD_out is routed to Da_TxRx when TXRXEN is disabled and to D_RxTx when TXRXEN is enabled. Similarly, D_RxTx is routed to TxD_in when TXRXEN is disabled and D_TxRx is routed to TxD_in when TXRXEN is enabled. The routing of frame syncs is shown in Figure 17-12 and the routing of interface clocks is shown in Figure 17-13.

If internal network mode is disabled, then RXDSEL selects the TxD_in, which is sent from the AUDMUX to the serial interface connected at Port x. When the internal network mode is selected, RxD_out is constructed by ANDing selected TxD_in signals from the ports (as determined by INMMASK).

If there is more than one device attached to the external port at D_TxRx and D_RxTx and one of the devices is a network master, then two conditions must be noted:

1. When the external master is enabled in network mode, then the serial interface at Port x must be configured as a slave (normal or network mode). No Tx/Rx switching is required.
2. When the external master is disabled and the serial interface at Port x and other slave devices must communicate, then the serial interface at Port x must be configured as a network mode master and the Tx/Rx switch at Port y must be enabled (TXRXEN = 1). This will ensure that the transmit and receive paths are connected appropriately.

To communicate with more than one port, internal network mode can be enabled at Port x. In internal network mode, it is possible to communicate with any device attached to the other ports. Internal network mode shall be enabled at the port that is the SSI network mode master.

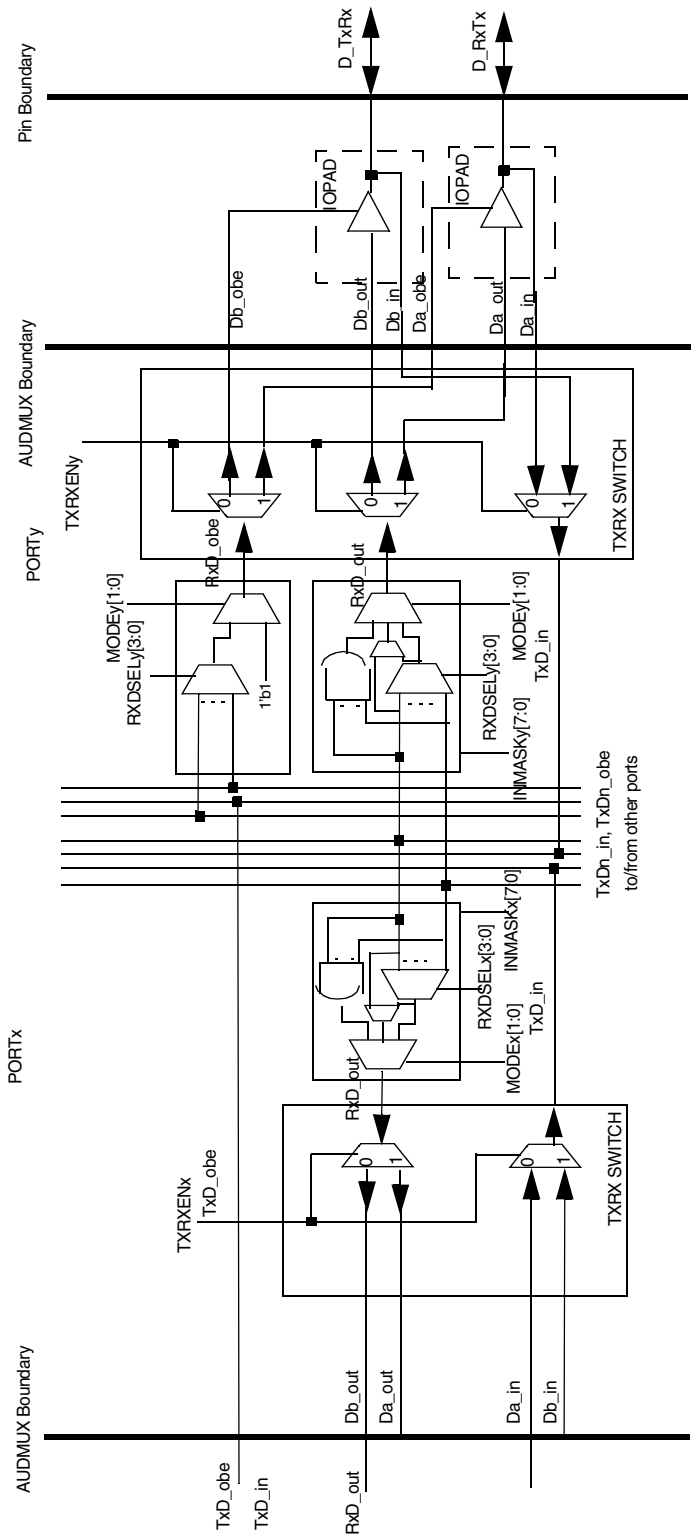


Figure 17-14. Internal to External Port Interconnection

17.1.3.2 External Port to External Port Connectivity

External ports can communicate with external ports directly. External ports can communicate together in three ways:

1. Each port's receive logic is configured in normal mode (MODE[0:1] = 00). Each port's RXDSEL[2:0] field is configured to select the other port's transmit data. Bit fields associated with clock/frame sync selection and direction are configured for each port. Either port can be the master.
2. One port is configured in internal network mode (MODE[0:1] = 01). All desired data lines are combined by the AND gate as determined by INMMASK[7:0]. Since an external port is being used as the internal network mode master, all other devices on the same AUDMUX port as the internal network mode master must be disabled. This configuration can be used with a combination of internal and external ports. All external ports must have a pull-up resistor on its RxDATA pin. Bit fields associated with clock/frame sync selection and direction are configured for each port. Any port can be the master.

17.1.3.3 Internal Port to Internal Port Connectivity

Internal ports can communicate with other internal ports directly. Internal ports can communicate together in three ways:

1. Each port's receive logic is configured in normal mode (MODE[0:1] = 00). Each port's RXDSEL[2:0] field is configured to select the other port's transmit data. Bit fields associated with clock/frame sync selection and direction are configured for each port. Either port can be the master.
2. One port is configured in internal network mode (MODE[0:1] = 01). All desired data lines are combined by the AND gate as determined by INMMASK[7:0]. This configuration can be used with a combination of internal and external ports. All external ports must have a pull-up resistor on its RxDATA pin. Bit fields associated with clock/frame sync selection and direction are configured for each port. Any port can be the master.

17.1.3.4 Loopback Connectivity

AUDMUX ports can communicate with themselves in order to provide loopback functionality. Port x can route its TxDATA signal to its own RxD_out signal by setting RXDSELx[2:0] to its own port number. This is supported by all ports in the AUDMUX.

In addition, ports can provide loopback support in internal network mode. With internal network mode, the internal network mode master can loop its TxDATA signal (combined with those of other ports, if desired) back into its RxD_out signal. Port x's INMMASK should be set such that bit (x - 1) is clear in order to enable the loopback.

17.2 External Signal Description

The following section provides the external signal descriptions for AUDMUX.

17.2.1 Overview

Table 17-1 lists pad-level signals of the AUDMUX for the external ports, where P*n* is P, to P7 and m = n -.The port is configured as an external port by a static system level signal input p*n*_int_ext_select.

Table 17-1. Signal Properties

Name	Port	I/O	Function	Reset State	Pull-up
ADm_TXD	P*n*	**I/O**	Transmitted Data from Pn	1	Active
ADm_RXD	P*n*	I/O	Received Data at Pn	1	Active
ADm_TXC	P*n*	I/O	Transmit Clock input/output at Pn	1	-
ADm_RXC	P*n*	I/O	Receive Clock input/output at Pn	1	-
ADm_TXFS	P*n*	I/O	Transmit Frame sync input/output at Pn	1	-
ADm_RXFS	P*n*	I/O	Receive Frame sync input/output at Pn	1	-

17.3 Memory Map and Register Definition

The AUDMUX memory map is shown in Table 17-2.

Table 17-2. AUDMUX Memory Map

Address	Register	Access	Reset Value	Section/Page
0x53FC_4000 (PTCR1)	Port Timing Control Register 1 (PTCR1)	RW	0xAD40_0800	17.3.2.1/17-26
0x53FC_4004 (PDCR1)	Port Data Control Register 1 (PDCR1)	RW	0x0000_A000	17.3.2.2/17-28
0x53FC_4008 (PTCR2)	Port Timing Control Register 2 (PTCR2)	RW	0xA500_0800	17.3.2.3/17-29
0x53FC_400C (PDCR2)	Port Data Control Register 2 (PDCR2)	RW	0x0000_8000	17.3.2.4/17-31
0x53FC_4010 (PTCR3)	Port Timing Control Register 3 (PTCR3)	RW	0x9CC0_0800	17.3.2.5/17-33
0x53FC_4014 (PDCR3)	Port Data Control Register 3 (PDCR3)	RW	0x0000_6000	17.3.2.6/17-35
0x53FC_4018 (PTCR4)	Port Timing Control Register 4 (PTCR4)	RW	0x0000_0800	17.3.2.7/17-36
0x53FC_401C (PDCR4)	Port Data Control Register 4 (PDCR4)	RW	0x0000_4000	17.3.2.8/17-38
0x53FC_4020 (PTCR5)	Port Timing Control Register 5 (PTCR5)	RW	0x0000_0800	17.3.2.9/17-40
0x53FC_4024 (PDCR5)	Port Data Control Register 5 (PDCR5)	RW	0x0000_2000	17.3.2.10/17-42
0x53FC_4028 (PTCR6)	Port Timing Control Register 6 (PTCR6)	RW	0x0000_0800	17.3.2.11/17-43
0x53FC_402C (PDCR6)	Port Data Control Register 6 (PDCR6)	RW	0x0000_0000	17.3.2.12/17-45

Table 17-2. AUDMUX Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0x53FC_400x30 (PTCR7)	Port Timing Control Register 7 (PTCR7)	RW	0x0000_0800	17.3.2.13/17-46
0x53FC_4034 (PDCR7)	Port Data Control Register 7 (PDCR7)	RW	0x0000_C000	17.3.2.14/17-48

17.3.1 Register Summary

Table 17-4 shows the control register and address mapping for the AUDMUX. Figure 17-5 shows the register fields key. Table 17-3 shows the register figure conventions.

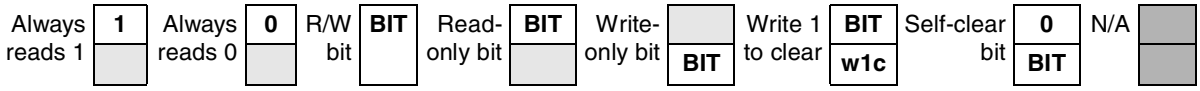


Figure 17-15. Key to Register Fields

Table 17-3. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 17-4. Register Summary

Address		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x53FC_4000 (PTCR1)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_4004 (PDCR1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																
0x53FC_4008 (PTCR2)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_400C (PDCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																
0x53FC_4010 (PTCR3)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_4014 (PDCR3)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																
	W																

Table 17-4. Register Summary (continued)

Address		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x53FC_4018 (PTCR4)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_401C (PDCR4)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																
0x53FC_4020 (PTCR5)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_4024 (PDCR5)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																
0x53FC_4028 (PTCR6)	R	TFSDIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFSDIR	RFSEL[3:0]				RCLKDIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_402C (PDCR6)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
	W																

Table 17-4. Register Summary (continued)

Address		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x53FC_400x30 (PTCR7)	R	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCLK DIR
	W																
	R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
	W																
0x53FC_4034 (PDCR7)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RXDSEL[2:0]			TXRX EN	0	0	0	MODE	INMMASK[7:0]							
	W																
	W																

17.3.2 Register Descriptions

There are two configuration registers for each port. There are a total of 15 configuration registers.

17.3.2.1 Port Timing Control Register 1 (PTCR1)

PTCR1 is the Port Timing Control Register for Port 1.

0x53FC_4000 (PTCR1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFS DIR	TFSEL[3:0]			TCLK DIR	TCSEL[3:0]			RFS DIR	RFSEL[3:0]						RCL KDIR
W																
Reset	1	0	1	0	1	1	0	1	0	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-16. Port Timing Control Register for Port 1

Table 17-5. Register Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.

Table 17-5. Register Field Descriptions (continued)

Field	Description
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.2 Port Data Control Register 1 (PDCR1)

PDCR1 is the Port Data Control Register for Port 1.

0x53FC_4004 (PDCR1) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXR XEN	0	0	0	MOD E	INMMASK[7:0]							
W																
Reset	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-17. Port Data Control Register for Port 1 (PDCR1)

Table 17-6. PDCR (Port 1) Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. RXDSEL is ignored if MODE[1:0] is 2'b01 (that is, Internal Network Mode is enabled). xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved

Table 17-6. PDCR (Port 1) Field Descriptions (continued)

Field	Description
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11-9	Reserved
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit 0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.3 Port Timing Control Register 2 (PTCR2)

PTCR2 is the Port Timing Control Register for Port 2.

0x53FC_4008 (PTCR2) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-18. Port Timing Control Register 2 (PTCR2)

Table 17-7. PTCR2 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL	RFSEL - Receive Frame Sync Select Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.

Table 17-7. PTCR2 Field Descriptions (continued)

Field	Description
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	SYN—Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.4 Port Data Control Register 2 (PDCR2)

PDCR2 is the Port Data Control Register for Port 2.

0x53FC_400C (PDCR2) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXR XEN	0	0	0	MOD E	INMMASK[7:0]							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-19. Port Data Control Register 2 (PDCR2)

Table 17-8. PDCR2 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. RXDSEL is ignored if MODE[1:0] is 01 (that is, Internal Network mode is enabled). xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11-9	Reserved
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDED together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.5 Port Timing Control Register 3 (PTCR3)

PTCR3 is the Port Timing Control Register for Port 3.

0x53FC_4010 (PTCR3) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	1	0	0	1	1	1	0	0	1	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-20. Port Timing Control Register 3 (PTCR3)

Table 17-9. PTCR3 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects RxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved

Table 17-9. PTCR3 Field Descriptions (continued)

Field	Description
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	SYN—Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.6 Port Data Control Register 3 (PDCR3)

PDCR3 is the Port Data Control Register for Port 3.

0x53FC_4014 (PDCR3) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
W																
Reset	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-21. Port Data Control Register 3 (PDCR3)

Table 17-10. PDCR3 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. RXDSEL is ignored if MODE[1:0] is 01 (that is, Internal Network Mode is enabled). xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11-9	Reserved

Table 17-10. PDCR3 Field Descriptions (continued)

Field	Description
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.7 Port Timing Control Register 4 (PTCR4)

PTCR4 is the Port Timing Control Register for Port 4.

0x53FC_4018 (PTCR4) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-22. Port Timing Control Register 4 (PTCR4)

Table 17-11. PTCR4 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects RxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.

Table 17-11. PTCR4 Field Descriptions (continued)

Field	Description
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	SYN—Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.8 Port Data Control Register 4 (PDCR4)

PDCR4 is the Port Data Control Register for Port 4.

0x53FC_401C (PDCR4) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXR XEN	0	0	0	MOD E	INMMASK[7:0]							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-23. Port Data Control Register 4 (PDCR4)

Table 17-12. PDCR4 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. RXDSEL is ignored if MODE[1:0] is 01 (that is, Internal Network Mode is enabled). xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11-9	Reserved
	Mode Select. This field selects the mode in which the port is to operate.The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDED together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.9 Port Timing Control Register 5 (PTCR5)

PTCR5 is the Port Timing Control Register for Port 5.

0x53FC_4020 (PTCR5) — Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-24. Port Timing Control Register 5 (PTCR5)

Table 17-13. PTCR5 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects RxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved

Table 17-13. PTCR5 Field Descriptions (continued)

Field	Description
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	SYN—Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.10 Port Data Control Register 5 (PDCR5)

0x53FC_4024 (PDCR5)

Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-25. Port Data Control Register 5 (PDCR5)

Table 17-14. PDCR5 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. RXDSEL is ignored if MODE[1:0] is 01 (that is, Internal Network Mode is enabled). xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11-9	Reserved
8 MODE	Mode Select. This field selects the mode in which the port is to operate.The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDED together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.11 Port Timing Control Register 6 (PTCR6)

PTCR6 is the Port Timing Control Register for Port 6.

0x53FC_4028 (PTCR6) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-26. Port Timing Control Register 6 (PTCR6)

Table 17-15. PTCR6 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects RxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved

Table 17-15. PTCR6 Field Descriptions (continued)

Field	Description
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	SYN—Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.12 Port Data Control Register 6 (PDCR6)

PDCR6 is the Port Data Control Register for Port 6.

0x53FC_402C (PDCR6) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-27. Port Data Control Register 6 (PDCR6)

Table 17-16. PDCR6 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9	Reserved
8 MODE	Mode Select. This field selects the mode in which the port is to operate.The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.2.13 Port Timing Control Register 7 (PTCR7)

PTCR7 is the Port Timing Control Register for Port 7.

0x53FC_400x30 (PTCR7) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TFS DIR	TFSEL[3:0]				TCLK DIR	TCSEL[3:0]				RFS DIR	RFSEL[3:0]				RCL KDIR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Figure 17-28. Port Timing Control Register 7 (PTCR7)

Table 17-17. PTCR7 Field Descriptions

Field	Description
31 TFS DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TxFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TxFS is an input. 1 TxFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TxFS is sourced. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TxClk pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock. 0 TxClk is an input. 1 TxClk is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TxClk is sourced. 0xxx Selects RxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved

Table 17-17. PTCR7 Field Descriptions (continued)

Field	Description
21 RFS DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RxFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync. 0 RxFS is an input. 1 RxFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RxFS is sourced. RxFS can be sourced from TxFS and RxFS from other ports. 0xxx Selects TxFS from port. 1xxx Selects RxFS from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RxClk pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock. 0 RxClk is an input. 1 RxClk is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RxClk is sourced. RxClk can be sourced from TxClk and RxClk from other ports. 0xxx Selects TxClk from port. 1xxx Selects RxClk from port. x000 Port 1 . . . x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface). 0 Asynchronous mode 1 Synchronous mode (default)
10–0	Reserved

17.3.2.14 Port Data Control Register 7 (PDCR7)

PDCR7 is the Port Data Control Register for Port 7.

0x53FC_4034 (PDCR7) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0	0	0	MODE	INMMASK[7:0]							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 17-29. Port Data Control Register 7 (PDCR7)

Table 17-18. PDCR7 Field Descriptions

Field	Description
31–16	Reserved
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RxD data. xxx Port number for RxD 000 Port 1 . . . 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9	Reserved
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: • Normal mode, in which the RxD from the port selected by RXDSEL is routed to the port. • Internal Network mode in which RxD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RxD signals are ANDed together. 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which of the RxD signals are to be ANDed together for internal network mode. Bit 6 represents RxD from Port 7 and bit0 represents RxD from Port 1. 0 Includes RxDn for ANDing 1 Excludes RxDn from ANDing

17.3.3 AUDMUX Default Configuration

The AUDMUX reverts back to its default settings following a reset. Section 17.3.3.1, Default Port Configuration," describe the default configuration of the ports.

17.3.3.1 Default Port Configuration

The AUDMUX's default port configuration after reset is as follows:

- Port 1 connected to Port 6
  - Port 6 provides the clock and frame sync.
  - Synchronous mode is enabled.
  - Normal mode is selected.
- Port 2 connected to Port 5
  - Port 5 provides the clock and frame sync.
  - Synchronous mode is enabled.
  - Normal mode is selected.
- Port 3 connected to Port 4
  - Port 4 provides the clock and frame sync.
  - Synchronous mode is enabled.
  - Normal mode is selected.
- Port 7 in data loopback mode
  - Clock and frame syncs are inputs.
  - Synchronous mode is enabled.
  - Normal mode is selected.

17.4 AUDMUX Clocking

This section provides information about AUDMUX clocking including clock inputs and the clock diagram.

17.4.1 AUDMUX Clock Inputs

The IP Bus read/write clock—ipg_clk_s—is an input to the AUDMUX. It is used for all AUDMUX register accesses. It is driven only when there is an AUDMUX access on the IP Bus.

) or external CODECs. The clock used for CE Bus Network mode is determined by PTCR7. Refer to Section Section , "CE_Bus_dis Signal Generation," for more details of the clock selection for CE Bus Network mode.

17.4.2 AUDMUX Clock Diagram

Figure 17-30 shows the clocking used in the AUDMUX.

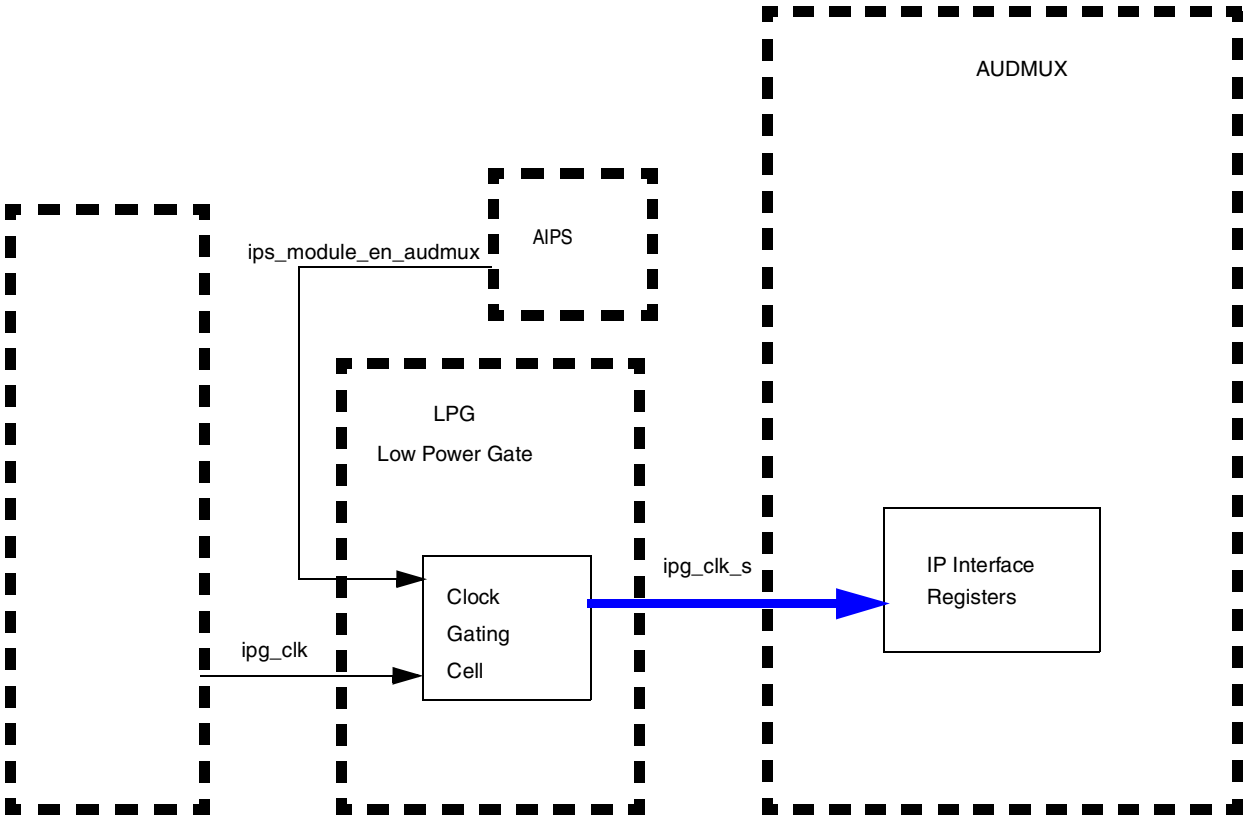


Figure 17-30. AUDMUX Clocking Scheme

17.4.3 Clocking Restrictions

- Since the AUDMUX requires only ipg_clk_s, the AUDMUX places no restrictions on the bus frequency.
- All registers in the AUDMUX are control registers so their values will not change frequently. Their values will be programmed when changing between use cases (not during use cases).

17.5 Initialization/Application Information

This section provides initialization and application information for AUDMUX.

Chapter 18
Configurable Serial Peripheral Interface (CSPI)

The Configurable Serial Peripheral Interface (CSPI) module allows rapid data communication with fewer software interrupts than conventional serial communications. The i.MX51 module contains one 8 × 32 receive buffer (RXFIFO) and one 8 × 32 transmit buffer (TXFIFO). Figure 18-1 shows the i.MX51 block diagram.

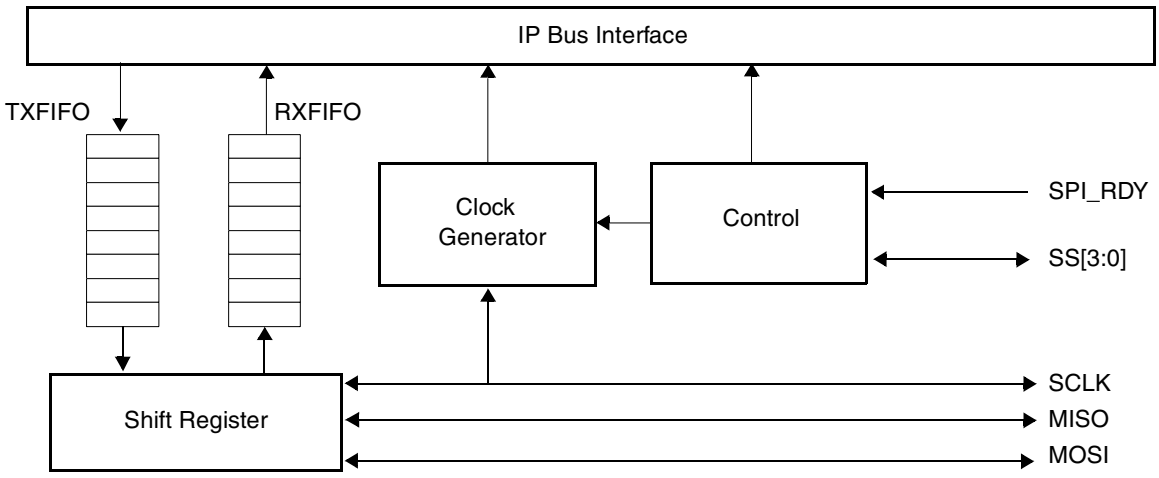


Figure 18-1. i.MX51 Block Diagram

18.1 Overview

The i.MX51 is equipped with data FIFOs and is a master/slave configurable serial peripheral interface module, capable of interfacing to both SPI master and slave devices. The i.MX51 Ready ($\overline{\text{SPI\_RDY}}$) and Chip Select ($\overline{\text{SS}}$) control signals enable fast data communication with fewer software interrupts.

This chapter describes the configuration and operation of the CSPI. Information such as base addresses and features that are unique to the multiple devices is described as appropriate.

18.1.1 Features

The i.MX51 is used for fast data communication with fewer software interrupts. It includes the following features:

- Full-duplex synchronous serial interface
- Master/Slave configurable
- Four chip selects to support multiple peripherals

- Transfer continuation function allows unlimited length data transfers
- 32-bit wide by 8-entry FIFO for both transmit and receive data
- Polarity and phase of the Chip Select ($\overline{SS}$) and SPI Clock (SCLK) are configurable
- DMA support

18.1.2 Modes of Operation

This module can be configured for master and slave modes. The details of each mode are as follows:

- Master Mode

  When the CSPI module is configured as a master, it uses a serial link to transfer data between the CSPI and an external device. A chip-enable signal and a clock signal are used to transfer data between these two devices. If the external device is a transmit-only device, the CSPI master's output port can be ignored and used for other purposes. In order to use the internal TXFIFO and RXFIFO, two auxiliary output signals, $\overline{SS}$ and $\overline{SPI\_RDY}$, are used for data transfer rate control. The user can also program the sample period control register to a fixed data transfer rate.

- Slave Mode

  When the CSPI module is configured as a slave, the user can configure the CSPI Control register to match the external SPI master's timing. In this configuration, $\overline{SS}$ becomes an input signal and is used to control data transfers through the Shift register as well as to load/store the data FIFO.

18.2 External Signal Description

The following signals shown in Table 18-1 are used to control the serial peripheral interface.

Table 18-1. CSPI Signal Properties

Name	Function	I/O	Reset	Pull-up
$\overline{SS}$[3:0]	Chip selects	I/O	1	—
SCLK	SPI clock	I/O	0	Active
MISO	Master data in; slave data out	I/O	0	Passive
MOSI	Master data out; slave data in	I/O	0	—
$\overline{SPI\_RDY}$	SPI data ready in Master mode	I	1	Active

Table 18-2 provides a detailed description of the CSPI signals.

Table 18-2. CSPI – Detailed Signal Descriptions

Signal	I/O	Description	
MOSI	I/O	Master Out Slave In. In Master mode, this bidirectional signal is a TX output signal from the Data Shift register. In Slave mode, it is an RX input from external SPI device.	
		State Meaning	Asserted—Indicates a 1 is transmitted. Negated—Indicates a 0 is transmitted.
		Timing	Asserted Negated
MISO	I/O	Master In Slave Out – In Master mode, this bidirectional signal is an RX input signal to the Data Shift register. In Slave mode, it is a TX output to external SPI device.	
		State Meaning	Asserted—Indicates a 1 is transmitted. Negated—Indicates a 0 is transmitted.
		Timing	Asserted Negated
SCLK	I/O	SPI Clock – In Master mode, this bidirectional signal is a SPI clock output. In Slave mode, it is a SPI clock input.	
		State Meaning	Asserted—Clock is high. Negated—Clock is low.
		Timing	Asserted Negated
$\overline{SS}$[3:0]	I/O	Chip selects – In Master mode, these bidirectional signals are outputs. In Slave mode, these are inputs. These signals are selected in/out by Chip Select bits in the CSPI Control register.	
		State Meaning	Asserted—When SSPOL is set, device is selected, otherwise device is de-selected. Negated—When SSPOL is set, device is de-selected, otherwise device is selected.
		Timing	Asserted Negated
$\overline{SPI\_RDY}$	I	SPI Ready – It is an input from an external SPI slave device. This signal triggers the CSPI to start a burst. Edge-trigger or level-trigger can be configured in the CSPI Control register.	
		State Meaning	Asserted—External SPI device is not ready. Negated—External SPI device is ready.
		Timing	Asserted Negated

18.3 Memory Map and Register Definition

The i.MX51 contains one CSPI module, which includes eight 32-bit registers. Section 18.3.3, Register Descriptions” provides the detailed descriptions for the CSPI registers.

18.3.1 Memory Map

Table 18-3 shows the CSPI memory map.

Table 18-3. i.MX51 CSPI Memory Map[1]

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x000 (RXDATA)	Receive Data Register (RXDATA)	Read-only	0x0000_0000	18.3.3.1/18-6
0xBASE+0x004 (TXDATA)	Transmit Data Register (TXDATA)	Write-only	0x0000_0000	18.3.3.2/18-7
0xBASE+0x008 (CONREG)	Control Register (CONREG)	R/W	0x0000_0000	18.3.3.3/18-8
0xBASE+0x00C (INTREG)	Interrupt Control Register (INTREG)	R/W	0x0000_0000	18.3.3.4/18-11
0xBASE+0x010 (DMAREG)	DMA Control Register (DMAREG)	R/W	0x0000_0000	18.3.3.5/18-12
0xBASE+0x014 (STATREG)	Status Register (STATREG)	R/W	0x0000_0003	18.3.3.6/18-13
0xBASE+0x018 (PERIODREG)	Sample Period Control Register (PERIODREG)	R/W	0x0000_0000	18.3.3.7/18-14
0xBASE+0x01C (TESTREG)	Test Control Register (TESTREG)	R/W	0x0000_0000	18.3.3.8/18-15

1 See Chapter 2, “Memory Map,” for the value of base address of CSPI.

18.3.2 Register Summary

Figure 18-2 shows the key to the register fields and Table 18-4 shows the register figure conventions.

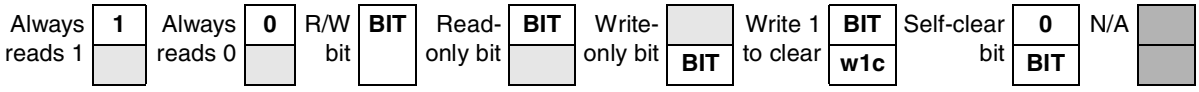


Figure 18-2. Key to Register Fields

Table 18-4. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit’s value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	

Table 18-4. Register Figure Conventions (continued)

Convention	Description
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 18-5 shows the i.MX51 register summary.

Table 18-5. CSPI Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (RXDATA)	R	RXDATA[31:16]															
	W																
	R	RXDATA[15:0]															
	W																
0xBASE+0x004 (TXDATA)	R																
	W	TXDATA[31:16]															
	R																
	W	TXDATA[15:0]															
0xBASE+0x008 (CONREG)	R	BURST LENGTH												0	DATA RATE		
	W																
	R	0	0	CHIP SELECT		0	0	DRCTL		SSP OL	SSC TL	PHA	POL	SM C	XCH	MO DE	EN
	W																
0xBASE+0x00C (INTREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	TCE N	ROE N	RFE N	RHE N	RRE N	TFE N	THE N	TEE N
	W																
0xBASE+0x010 (DMAREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	RF DEN	RH DEN	0	0	TH DEN	TE DEN
	W																

Table 18-5. CSPI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x014 (STATREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	TC	RO	RF	RH	RR	TF	TH	TE
	W									w1c							
0xBASE+0x018 (PERIODREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	CSRC	SAMPLE PERIOD[14:0]														
	W																
0xBASE+0x01C (TESTREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	SWAP	LBC	0	0	SMSTATUS				RXCNT				TXCNT			
	W																

18.3.3 Register Descriptions

The following section describes the detailed register descriptions for the CSPI registers.

18.3.3.1 Receive Data Register (RXDATA)

The Receive Data register (RXDATA) is a read-only register that forms the top word of the 8×32 receive FIFO. This register holds the data received from an external SPI device during a data transaction. Only word-sized read operations are allowed. Figure 18-3 shows the RXDATA register and Table 18-6 shows the register's field descriptions.

0xBASE+0x000 (RXDATA) Access: User read-only

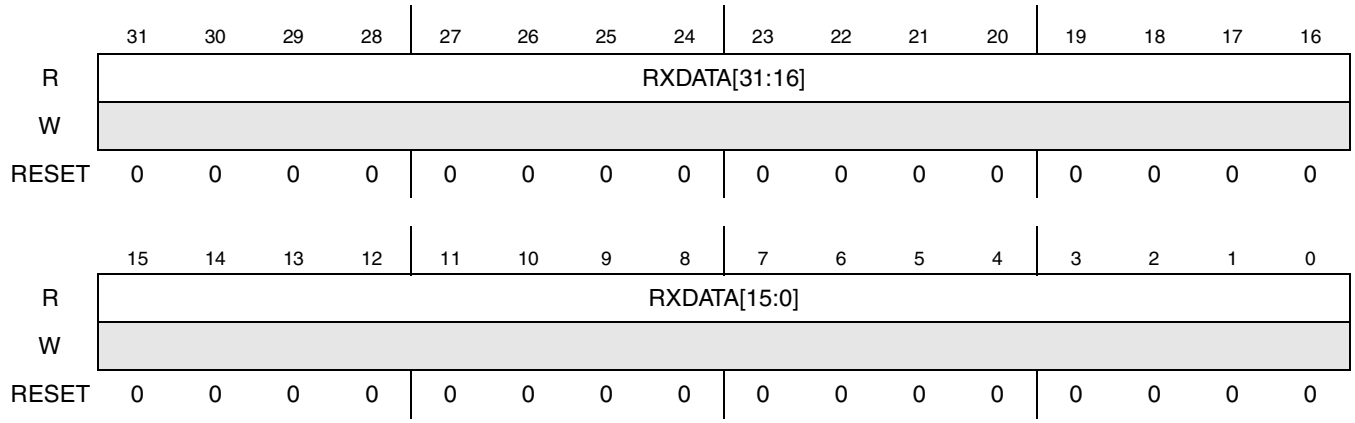


Figure 18-3. RXDATA Register Diagram

Table 18-6. RXDATA Register Field Description

Field	Description
31–0 RXDATA	Receive Data This register holds the top word of the receive data FIFO. The FIFO is advanced for each read of this register. The data read is undefined when the Receive Data Ready (RR) bit in the Interrupt Control/Status register is cleared. Zeros are read when CSPI is disabled.

18.3.3.2 Transmit Data Register (TXDATA)

The Transmit Data (TXDATA) register is a write-only data register that forms the top word of the 8×32 TXFIFO. The TXFIFO can be written to as long as it is not full, even when the XCH bit in CONREG is set. This allows the user write access to the TXFIFO during a SPI data exchange process. Writes to this register are ignored when the i.MX51 module is disabled (EN bit of i.MX51 CONREG is cleared). Figure 18-4 shows the CNTRL register and Table 18-7 shows the register's field descriptions.

0xBASE+0x004 (TXDATA) Access: User write-only

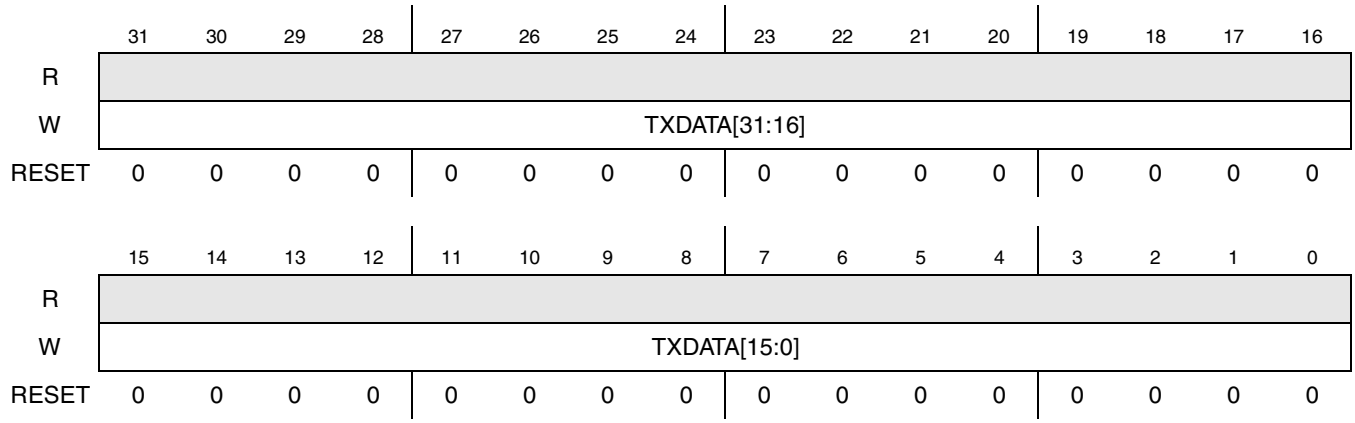


Figure 18-4. TXDATA Register Diagram

Table 18-7. TXDATA Register Field Description

Field	Description
31–0 TXDATA	Transmit Data This register holds the top word of data loaded into the FIFO. Data written to this register must be a word operation. The number of bits actually transmitted is determined by the BIT_COUNT field of the corresponding SPI Control register. If this field contains more bits than the number specified by BIT_COUNT, the extra bits are ignored. For example, to transfer 10 bits of data, a 32-bit word must be written to this register. Bits 9-0 are shifted out and bits 31-10 are ignored. When the CSPI module is operating in Slave mode, zeros are shifted out when the FIFO is empty. Zeros are read when CSPI is disabled.

18.3.3.3 Control Register (CONREG)

The Control Register (CONREG) allows the user to enable the i.MX51 module, configure its operating modes, specify the divider value, phase, and polarity of the clock, configure the $\overline{SS}$ and $\overline{SPI\_RDY}$ control signal, and define the transfer length. The reserved bits are always read as 0. Figure 18-5 shows the CNTRL register and Table 18-8 shows the register's field descriptions.

0xBASE+0x008 (CONREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BURST LENGTH												0	DATA RATE		
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	CHIP SELECT		0	0	DRCTL		SSPOL	SSCTL	PHA	POL	SMC	XCH	MODE	EN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18-5. CSPI Control Register

Table 18-8. CONREG Register Field Description

Field	Description
31–20 BURST LENGTH	Burst Length. This field defines the length of a SPI burst to be transferred. $\overline{SS}$ remains asserted until all bits in a SPI burst are shifted out. A maximum of 2^12 bits can be transferred in a single SPI burst. In master mode, it controls the number of bits per SPI burst. Since the shift register always loads 32-bit data from transmit FIFO, only the n least-significant (n = BURST LENGTH + 1) are shifted out. The remaining bits are ignored. In slave mode, only when SSCTL is cleared, this field takes effect in SPI transfer. Number of Valid Bits in a SPI burst. 0x000 A SPI burst contains least 1 bit in a word. 0x001 A SPI burst contains least 2 bit in a word. 0x010 A SPI burst contains least 3 bit in a word. ..... 0x01F A SPI burst contains all 32 bits in a word. 0x020 A SPI burst contains least 1 bit in first word and all 32 bits in second word. 0x021 A SPI burst contains least 2 bit in first word and all 32 bits in second word. ..... 0xFFE A SPI burst contains least 31 bits in first word and 2^7 -1 words. 0xFFF A SPI burst contains 2^7 words.
19	Reserved, all bits should read zero.
18–16 DATA RATE	SPI Data Rate Control This three-bit field selects the baud rate of the SCLK based on a division of the ipg_clk. These bits allow CSPI to synchronize with different external SPI devices. The max frequency is one quarter of ipg_clk. The divide ratio is determined according to the following table using the equation: $2^{(n+2)}$. SPI Data Rate Control (Master Mode only) 000 Divide by 4. 001 Divide by 8. 010 Divide by 16. 011 Divide by 32. 100 Divide by 64. 101 Divide by 128. 110 Divide by 256. 111 Divide by 512.
15 –14	Reserved, all bits should read zero.
13 –12 CHIP SELECT	CHIP SELECT Selects one of four external SPI Master/Slave Devices. In master mode, these two bits select the external slave devices by asserting the $\overline{SS}$n outputs. Only the selected $\overline{SS}$n signal are active while the remaining 3 signals are negated. Chip Select 00 $\overline{SS}$0 is asserted. 01 $\overline{SS}$1 is asserted. 10 $\overline{SS}$2 is asserted. 11 $\overline{SS}$3 is asserted.
11–10	Reserved, all bits should read zero.

Table 18-8. CONREG Register Field Description (continued)

Field	Description
9–8 DRCTL	SPI Data Ready Control This 2-bit field selects the utilization of the $\overline{\text{SPI\_RDY}}$ in master mode. CSPI checks this fields before it start a SPI burst. 00 Don't care $\overline{\text{SPI\_RDY}}$ 01 Burst are triggered by failing edge of $\overline{\text{SPI\_RDY}}$. 10 Burst are triggered by low level of $\overline{\text{SPI\_RDY}}$. 11 $\overline{\text{RSV}}$.
7 SSPOL	SPI SS Polarity Select In both Master and Slave mode, this bit selects the polarity of the $\overline{\text{SS}}$ signal. 0 Active low. 1 Active high.
6 SSCTL	SPI SS Wave Form Select In master mode, this bit controls the output wave form of $\overline{\text{SS}}$ signal when SMC is cleared, and it is ignored when SMC is set. 0 Only one SPI bursts is transmitted. 1 Negate $\overline{\text{SS}}$ between SPI bursts. Multiple SPI bursts are transmitted. CSPI transfer is stopped when TXFIFO is empty. In slave mode, this bit controls when a SPI burst is completed. 0 A SPI burst is completed when number of bits received in shifter register is equal to BURST LENGTH + 1. Only n least-significant bits (n = BURST LENGTH[4:0] + 1) of first received word are valid. And all bits in following words received in RXFIFO are valid. 1 A SPI burst is completed by $\overline{\text{SS}}$ edges. (SSPOL = 0: rising edge; SSPOL = 1: falling edge) RXFIFO is advanced whenever a $\overline{\text{SS}}$ edge is detected or shifter register contains 32-bits valid data.
5 PHA	SPI Clock/Data Phase Contro This bit controls the clock/data phase relationship. Refer to Figure 18-18, SPI Burst with Different POL and PHA Configuration, for description. 0 Phase 0 operation. 1 Phase 1 operation.
4 POL	SPI Clock Polarity Control This bit controls the polarity of the SCLK signal. Refer to Figure 18-18, SPI Burst with Different POL and PHA Configuration, for description. 0 Active high polarity (0 = Idle). 1 Active low polarity (1 = Idle).
3 SMC	Start Mode Control This bit is used in master mode only and it controls how CSPI start a SPI burst. 0 XCH bit controls when a SPI burst can start. Write a 1 to XCH bit starts a SPI burst or multiple bursts. (controlled by SSCTL) 1 Immediately start a SPI burst when data is written in TXFIFO.
2 XCH	SPI Exchange Bit If the SMC bit is cleared, writing a 1 to this bit starts one SPI bursts/multiple SPI bursts according to SSCTL bit. This bit remains set while either the exchange is in progress, or the i.MX51 is waiting for active input if $\overline{\text{SPIRDY}}$ is enabled through DRCTL. This bit is cleared automatically when all data in the TXFIFO and Shift register have been shifted out. In Slave mode, this bit is ignored. 0 Idle. 1 Initiates exchange (write) or busy (read).

Table 18-8. CONREG Register Field Description (continued)

Field	Description
1 MODE	SPI Function Mode Selec This bit selects the operating mode of the i.MX51. 0 Slave Mode. 1 Master Mode.
0 EN	SPI Module Enable Control This bit enables the i.MX51. This bit must be asserted before writing to other registers or initiating an exchange. Writing zero to this bit disables the module and resets the internal logic with the exception of the CONREG. The module's internal clocks are gated off whenever the module is disabled. 1 i.MX51 is enabled. 0 i.MX51 is disabled.

18.3.3.4 Interrupt Control Register (INTREG)

The 32-bit Interrupt Control Register (INTREG) enables the generation of interrupts to the MCU. The reserved bits cannot be written and always read as 0. If CSPI is disabled, this register reads zero. Figure 18-6 shows the CNTRL register and Figure 18-9 shows the register's field descriptions.

0xBASE+0x00C (INTREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	TCEN	ROEN	RFEN	RHEN	RREN	TFEN	THEN	TEEN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18-6. Interrupt Control Register Diagram

Table 18-9. INTREG Register Field Descriptions

Field	Description
31–8	Reserved, all bits should read zero.
7 TCEN	Transfer Completed Interrupt Enable. This bit enables the Transfer Completed Interrupt. 0 Disable 1 Enable
6 ROEN	RXFIFO Overflow Interrupt enable. The bit enables the RXFIFO Overflow Interrupt. 0 Disable 1 Enable

Table 18-9. INTREG Register Field Descriptions (continued)

Field	Description
5 RFEN	RXFIFO Full Interrupt enable. The bit enables the RXFIFO Full Interrupt. 0 Disable 1 Enable
4 RHEN	RXFIFO Half Full Interrupt enable. The bit enables the RXFIFO Half Full Interrupt. 0 Disable 1 Enable
3 RREN	RXFIFO Ready Interrupt enable. The bit enables the RXFIFO Ready Interrupt. 0 Disable 1 Enable
2 TFEN	TXFIFO Full Interrupt enable. The bit enables the TXFIFO Full Interrupt. 0 Disable 1 Enable
1 THEN	TXFIFO Half Empty Interrupt enable. The bit enables the TXFIFO Half Empty Interrupt. 0 Disable 1 Enable
0 TEEN	TXFIFO Empty Interrupt enable. The bit enables the TXFIFO Empty Interrupt. 0 Disable 1 Enable

18.3.3.5 DMA Control Register (DMAREG)

The DMA Control Register (DMAREG) provides the user a way to use the CSPI in DMA. Direct Memory Access (DMA) allows transfer of data between device and memory. Peripherals such as the CSPI supporting DMA use DMA request and acknowledge signals. The CSPI sends out DMA requests when the appropriate FIFO conditions are matched. The reserved bits cannot be written to and are always read as 0. If the CSPI is disabled, this register is also read as 0. Figure 18-7 shows the CNTRL register and Figure 18-10 shows the register's field descriptions.

0xBASE+0x010 (DMAREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	RF DEN	RH DEN	0	0	TH DEN	TE DEN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18-7. DMA Control Register Diagram

Table 18-10. DMAREG Register Field Description

Field	Description
31–6	Reserved, all bits should read zero.
5 RFDEN	RXFIFO Full DMA Request Enable. This bit enables/disables the RXFIFO Full DMA Request. 0 Disable 1 Enable
4 RHDEN	RXFIFO Half Full DMA Request Enable. This bit enables/disables the RXFIFO Half Full DMA Request. 0 Disable 1 Enable
3–2	Reserved, should be cleared.
1 THDEN	TXFIFO Half Empty DMA Request Enable. This bit enables/disables the TXFIFO Half Empty DMA Request. 0 Disable 1 Enable
0 TEDEN	TXFIFO Empty DMA Request Enable. This bit enables/disables the TXFIFO Empty DMA Request. 0 Disable 1 Enable

18.3.3.6 Status Register (STATREG)

The i.MX51 Status Register (STATREG) reflects the status of the CSPI module operating condition. The reserved bits cannot be written and always read as 0. If the CSPI is disabled, this register reads 0x0000_0003. Figure 18-8 shows the CNTRL register and Figure 18-11 shows the register's field descriptions.

0xBASE+0x014 (STATREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	TC	RO	RF	RH	RR	TF	TH	TE
W									w1c							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 18-8. Status Register Diagram

Table 18-11. STATREG Register Field Description

Field	Description
31–8	Reserved, should be cleared.
7 TC	Transfer Completed. When set, this bit indicates that all the data in TXFIFO has been loaded in the Shift register and Shift register has shifted out all the bits. Writing 1 to this bit clears it. 0 Busy. 1 Transfer Completed.
6 RO	RXFIFO Overflow. When set, this bit indicates that RXFIFO has overflowed. 0 RXFIFO is available. 1 RXFIFO has overflowed.
5 RF	RXFIFO Full. This bit is set when the RXFIFO is full (8 words). 0 Not Full. 1 Full.
4 RH	RXFIFO Half Full. This bit is set if the RXFIFO is half full (≥ 4 words in RXFIFO). 0 Less than 4 words are stored in RXFIFO. 1 Four or more words are available in RXFIFO.
3 RR	RXFIFO Ready. This bit is set any time there is one or more words stored in RXFIFO (≥ 1 words). 0 No valid data in RXFIFO. 1 More than 1 word in RXFIFO.
2 TF	TXFIFO Full. This bit is set when if the TXFIFO is full (8 words). 0 TXFIFO is not Full. 1 TXFIFO is Full.
1 TH	TXFIFO Half empty. This bit is set if the TXFIFO is more than half empty (≤ 4 words in TXFIFO). 0 TXFIFO holds more than 4 words. 1 TXFIFO holds 4 or fewer words.
0 TE	TXFIFO Empty. This bit is set if the TXFIFO is empty. 0 TXFIFO contains one or more words. 1 TXFIFO is empty.

18.3.3.7 Sample Period Control Register (PERIODREG)

The Sample Period Control Register (PERIODREG) provides the user a way to insert delays (wait states) between consecutive SPI transfers. Control bits in this register select the clock source for the sample period counter and the delay count indicating the number of wait states to be inserted between data transfers. Delay counts are only applicable when the i.MX51 module is operating in master mode. Figure 18-9 shows the CNTRL register and Figure 18-12 shows the register's field descriptions.

0xBASE+0x018 (PERIODREG) Access: User read/write

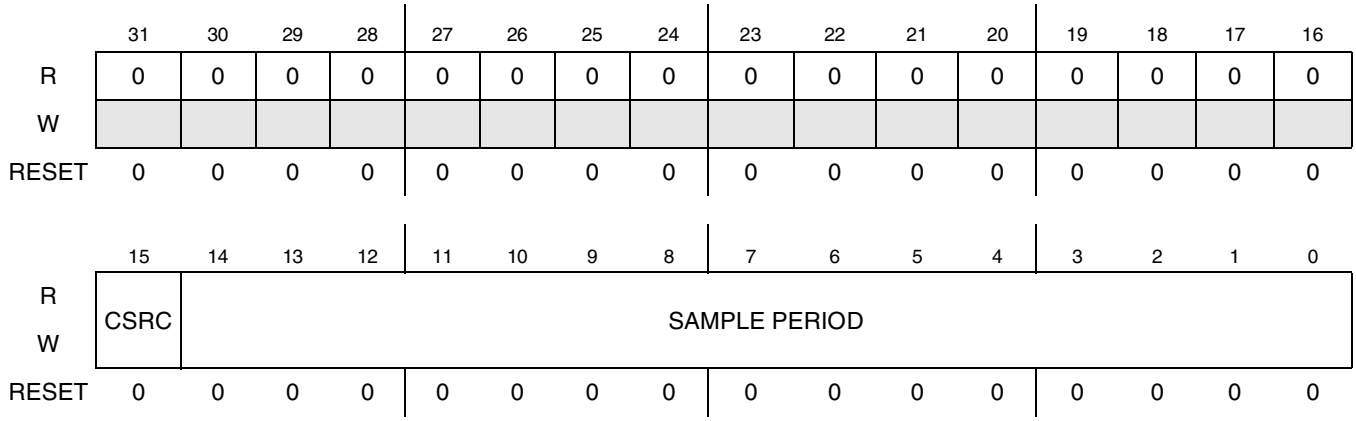


Figure 18-9. Sample Period Control Register Diagram

Table 18-12. PERIODREG Register Field Description

Field	Description
31–16	Reserved, all bits should read zero.
15 CSRC	Clock Source Control. This bit selects the clock source for the sample period counter. 0 SPI Clock (SCLK) 1 CKIL (32.768 KHz)
14–0 SAMPLE PERIOD	Sample Period Control. These bits control the number of wait states to be inserted in data transfers. During the idle clocks, the state of the $\overline{SS}$ output operates according to the SSCTL control field in CONREG. 0x0000 0 wait states inserted 0x0001 1 wait state inserted .... 0x7FFE 32766 wait states inserted 0x7FFF 32767 wait states inserted

18.3.3.8 Test Control Register (TESTREG)

The Test Control Register (TESTREG) provides the user a mechanism to internally connect the receive and transmit devices of the CSPI module, display the status of the state machine, monitor the contents of the receive and transmit FIFO, and debug the i.MX51. Figure 18-10 shows the CNTRL register and Figure 18-13 shows the register's field descriptions.

0xBASE+0x01C (TESTREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SWAP	LBC	0	0	SMSTATUS				RXCNT				TXCNT			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18-10. Test Control Register Diagram

Table 18-13. TESTREG Register Field Description

Field	Description
31–16	Reserved, All bits should be read as zero.
15 SWAP	Data Swap. This bit is used to swap data as it is read from the RXFIFO. When this bit is set, data read from RXFIFO is swapped. RXDATA[31:0] is swapped as follows: {RXDATA[7:0],RXDATA[15:8], RXDATA[23:16],RXDATA[31:24]} 0 Data read from RXFIFO is unchanged. 1 Data read from RXFIFO is swapped.
14 LBC	Loop Back Control. This bit is used in Master mode only. When this bit is set, the CSPI module connects the transmitter and receiver sections internally, and the data shifted out from the most-significant bit of the Shift register is looped back into the least-significant bit of the Shift register. In this way, a self-test of the complete transmit/receive path can be made. The output pins are not affected, and the input pins are ignored. 0 Not connected. 1 Internally connected.
13–12	Reserved, all bits should read zero.
11–8 SMSTATUS	State Machine Status. These bits indicate status of the state machine for test purpose.

Table 18-13. TESTREG Register Field Description (continued)

Field	Description
7–4 RXCNT	RXFIFO Counter. These bits indicate the number of words in RXFIFO. RXFIFO Counter 0000 0 word in RXFIFO 0001 1 word in RXFIFO .... 0111 7 words in RXFIFO 1000 8 words in RXFIFO
3–0 TXCNT	TXFIFO Counter. These bits indicate the number of words in TXFIFO. TXFIFO Counter 0000 0 word in TXFIFO 0001 1 word in TXFIFO .... 0111 7 words in TXFIFO 1000 8 words in TXFIFO

18.4 Functional Description

This section describes the timings for the CSPI. Figure 18-11 shows the generic CSPI timing.

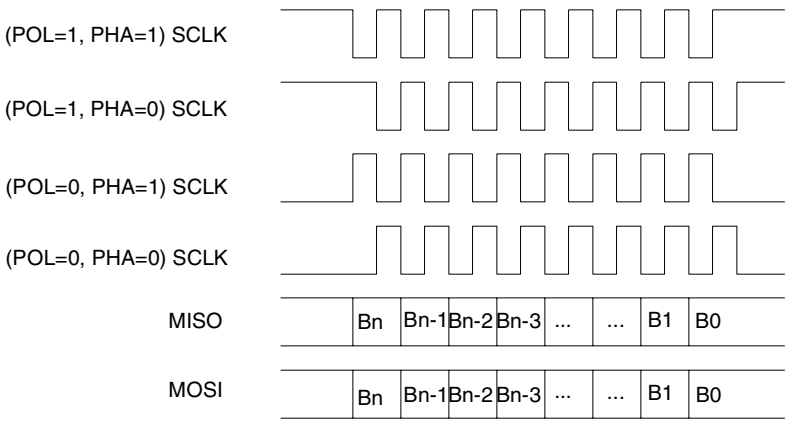


Figure 18-11. CSPI Generic Timing

18.4.1 Master Mode

The i.MX51 master uses the $\overline{SS}$ signal to enable an external SPI device and uses SPICLK to transfer data in and out of the Shift register. The $\overline{SPI\_RDY}$ enables fast data communication with fewer software interrupts. By using PERIODREG, the CSPI can be used for a fixed data transfer rate.

When CSPI is in Master mode the $\overline{SS}$, SCLK, and MOSI are output signals and the MISO is an input. Figure 18-12 shows a typical SPI burst.

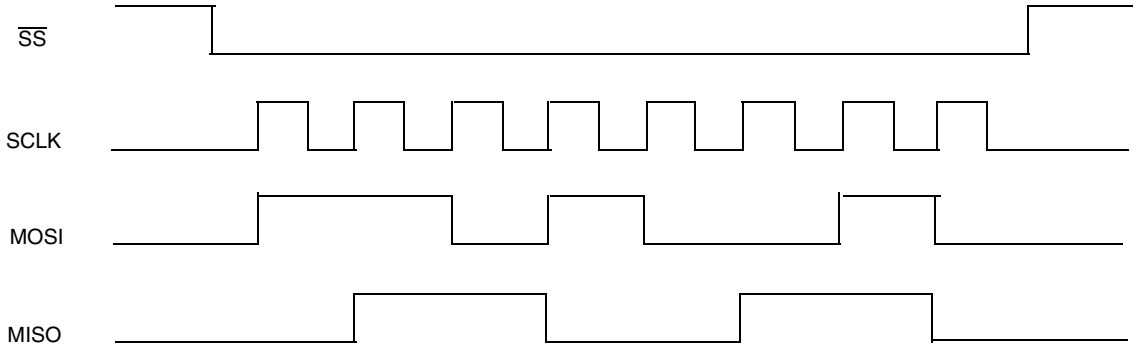


Figure 18-12. Typical SPI Burst (8-bit transfer)

In Figure 18-12, the SS signal enables the selected external SPI device and the SCLK synchronizes data transfer. MOSI and MISO change on rising edge of SCLK and the MISO is latched on the falling edge of the SLCK clock. The data shifted out is 0xD2, and the data shifted in is 0x66.

18.4.1.1 Master Mode with SPI_RDY

By default, the CSPI does not use SPI_RDY in master mode. a SPI burst begins when the following events happen: the CSPI is enabled, TXFIFO has data in it, and CONREG[XCH] or CONREG[SMC] is set. When CONREG[DRCTL] contains either 01 or 10, the SPI_RDY controls when a SPI burst starts.

If CONREG[DRCTL] is set to 01, the SPI burst can be triggered only if a falling edge of SPI_RDY has been detected. Figure 18-13 shows the relationship between a SPI burst and the falling edge of SPI_RDY.

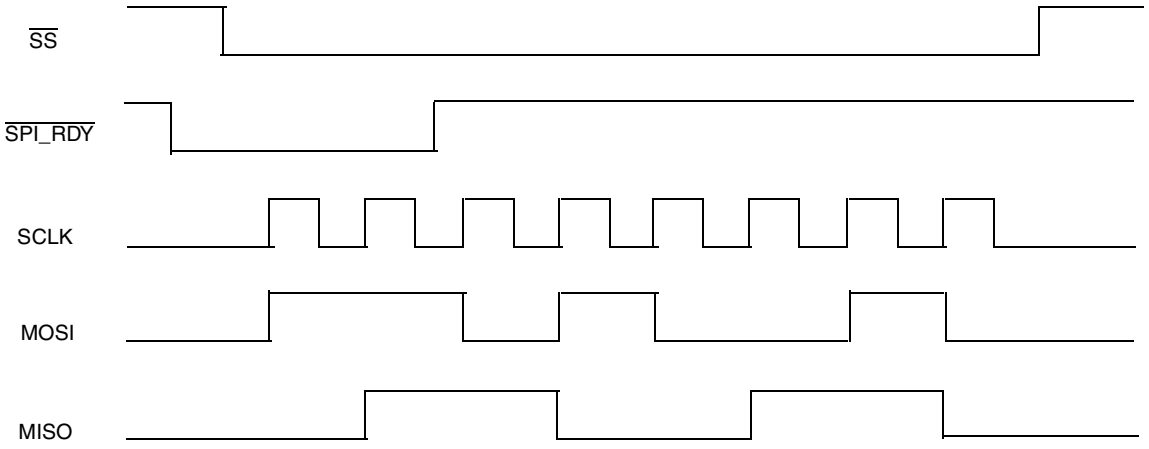


Figure 18-13. Relationship between a SPI Burst and the Falling Edge of SPI_RDY

A SPI burst does not start until the falling edge of SPI_RDY is detected. The next SPI burst starts when the next SPI_RDY falling edge is detected, after the last burst has finished.

If CONREG[DRCTL] is set to 10, the SPI burst can be triggered only if $\overline{\text{SPI\_RDY}}$ is low. Figure 18-14 shows the relationship between a SPI burst and $\overline{\text{SPI\_RDY}}$. The SPI burst does not begin until $\overline{\text{SPI\_RDY}}$ goes low. The next SPI burst begins after the last burst has finished if $\overline{\text{SPI\_RDY}}$ remains low.

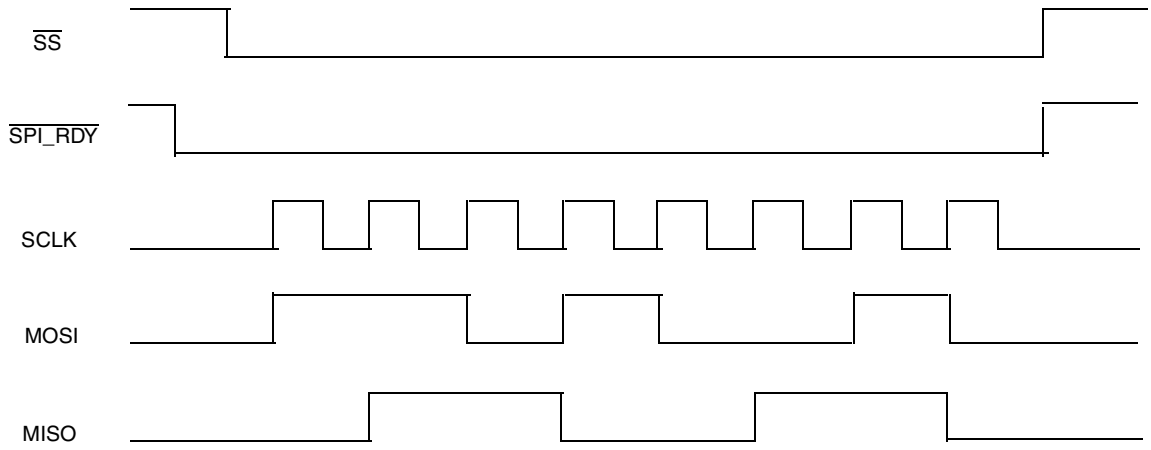


Figure 18-14. Relationship between a SPI Burst and $\overline{\text{SPI\_RDY}}$

18.4.1.2 Master Mode with Wait States

Wait states can be inserted between SPI bursts. This provides a way for the user to slow down the SPI burst to meet the timing requirements of a slower SPI device. Figure 18-15 shows wait states inserted between SPI bursts.

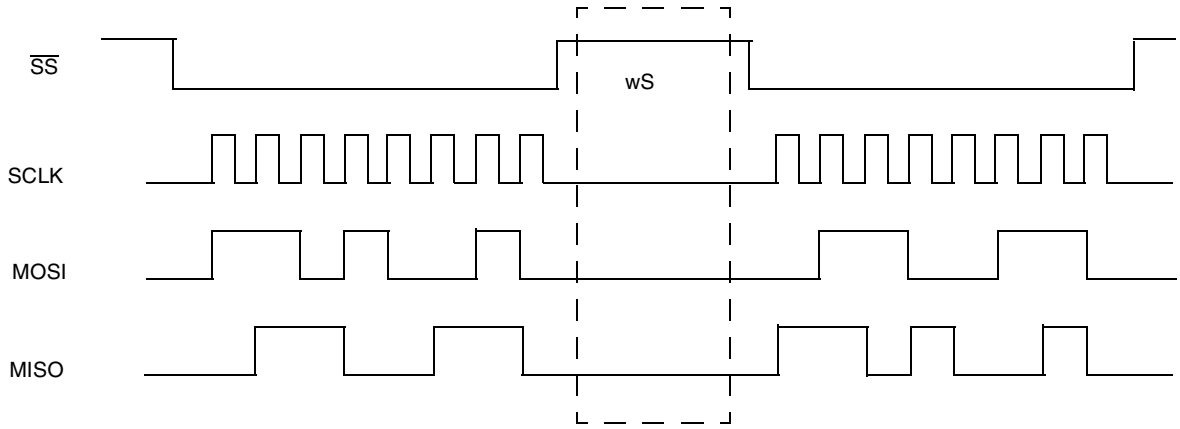


Figure 18-15. SPI Bursts with Wait States

In this case, the number of wait states is controlled by PERIODREG[SAMPLE PERIOD] and the wait states' clock source is selected by PERIODREG[CSRC].

18.4.1.3 Master Mode with SSCTL Control

SSCTL controls whether current operation is single burst or multiple bursts. When SSCTL is set, current operation is multiple bursts transfer. When SSCTL is cleared, current operation is single burst transfer. A SPI burst can contains multiple words as defined in BURST LENGTH.

Figure 18-16 shows one SPI burst while SSCTL is clear.

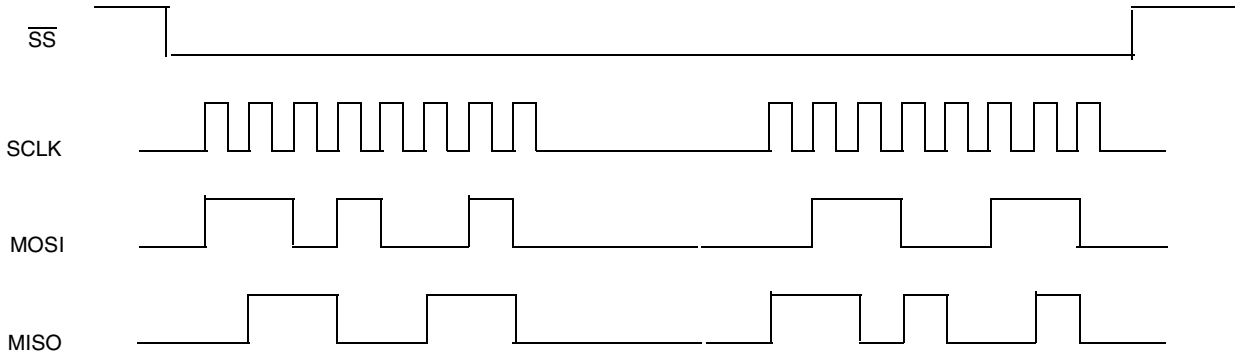


Figure 18-16. SPI Burst while SSCTL is Clear

In this case, two words in RXFIFO have been combined and transmitted in one SPI burst. The maximum length of a single SPI burst is defined in BURST LENGTH field in CONREG. This provides a way for transferring a longer SPI burst by writing data into TXFIFO while CSPI is transmitting.

Figure 18-17 shows two SPI bursts are transmitted while SSCTL is set.

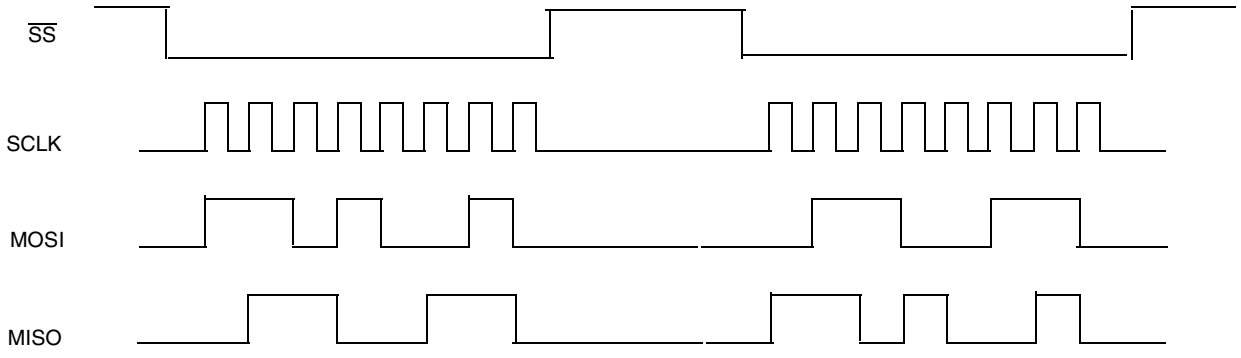


Figure 18-17. SPI Bursts while SSCTL is Set

In this case, two words are transmitted, one word per SPI burst. The CSPI continues transmitting SPI bursts until the TXFIFO is empty. When wait states can be inserted between SPI bursts, the $\overline{SS}$ negates between SPI bursts until the wait states finish.

18.4.1.4 Master Mode with PHA Control

CONREG[PHA] controls how the transmit data shifts out and the receive data shifts in.

When CONREG[PHA] is set, the transmit data shifts out on the rising edge of SCLK, and the receive data is latched on the falling edge of SCLK. The most-significant bit is output on the first rising SPICLK edge.

When CONREG[PHA] is cleared, the transmit data is shifted out on the falling edge of SCLK and the receive data is latched on the rising edge of SCLK. The most-significant bit is output when the CPU loads the transmitted data.

Inverting the SPICLK polarity does not impact the edge-triggered operations because they are internal to the serial peripheral interface master. Figure 18-18 shows a SPI burst using different POL and PHA configurations.

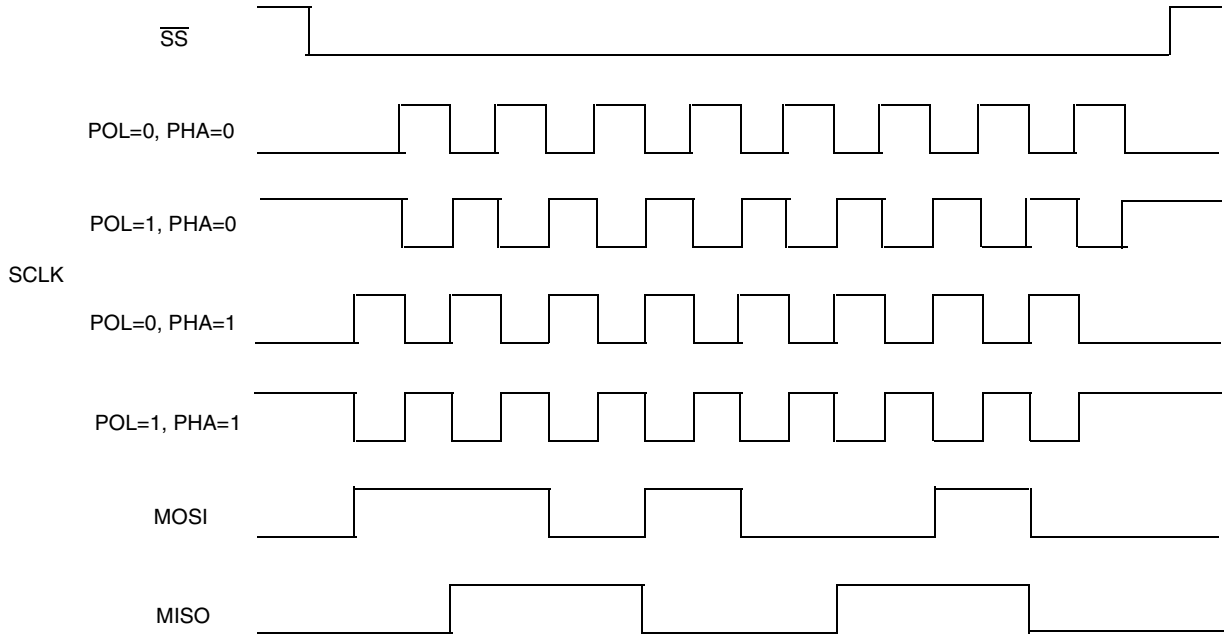


Figure 18-18. SPI Burst with Different POL and PHA Configuration

18.4.2 Slave Mode

When the CSPI module is configured as a slave, the user can configure the CSPI Control register to match the external SPI master's timing. In this configuration, $\overline{SS}$ becomes an input signal, and is used to latch data into or load data out to the internal data Shift registers, as well as to increment the data FIFO.

The $\overline{SS}$, SCLK, and MOSI are inputs and MISO is output. Most of their timing diagrams are the same as in Master mode, because the inputs come from a SPI master device.

However, it is different when $\overline{SS}$ is used to increment data FIFO. When the SSCTL is set while CSPI is in Slave mode, the data FIFO increments at $\overline{SS}$ rising edge (when SSPOL = 1, should be falling edge).

Figure 18-19 shows a SPI burst in which data FIFO is incremented by $\overline{SS}$ rising edge.

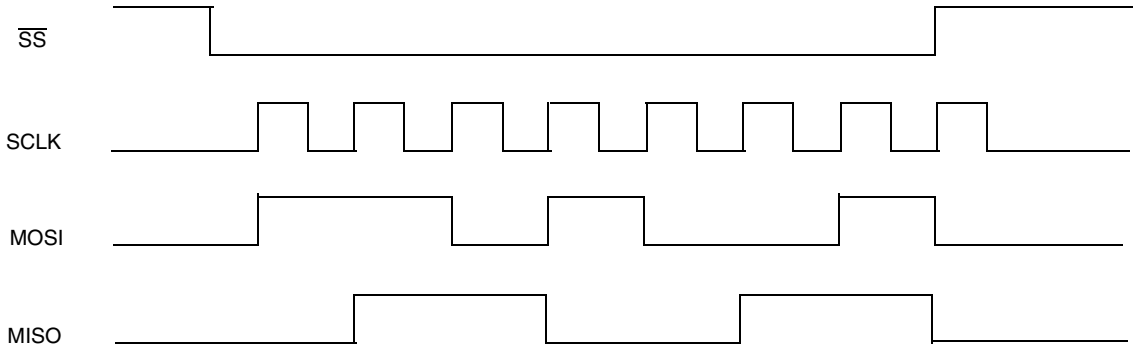


Figure 18-19. Increment Data FIFO by $\overline{SS}$ Rising Edge

In this case, the data received is not 0xD2 but 0x69. Only the most significant 7 bits are loaded to RXFIFO.

18.4.3 Interrupt Control

Interrupt control is not a specific mode of operation; however, it provides a basic method to utilize the CSPI FIFOs.

You can program the CSPI to enable the TXFIFO empty, TXFIFO half, and TXFIFO full interrupt. You can also use the interrupt service routine to fill the TXFIFO with data to be transferred. Furthermore, you can also enable RXFIFO ready, RXFIFO half, and RXFIFO full to retrieve data from RXFIFO by using the interrupt service routine.

Three other interrupt sources can be used to control/debug the SPI bursts. The transfer-completed interrupt tells the user that there is not data left in TXFIFO and the data in the Shift register is shifted out. The bit counter overflow interrupt tells the user that the CSPI received more than 32 bits in a SPI burst and the remaining bits will be lost. (Only in Slave mode does using $\overline{SS}$ increment the data FIFO.) The RXFIFO

overflow interrupt tells the user that the RXFIFO received more than 8 words and will not accept any other word. Figure 18-20 shows a program sequence of SPI bursts using interrupt.

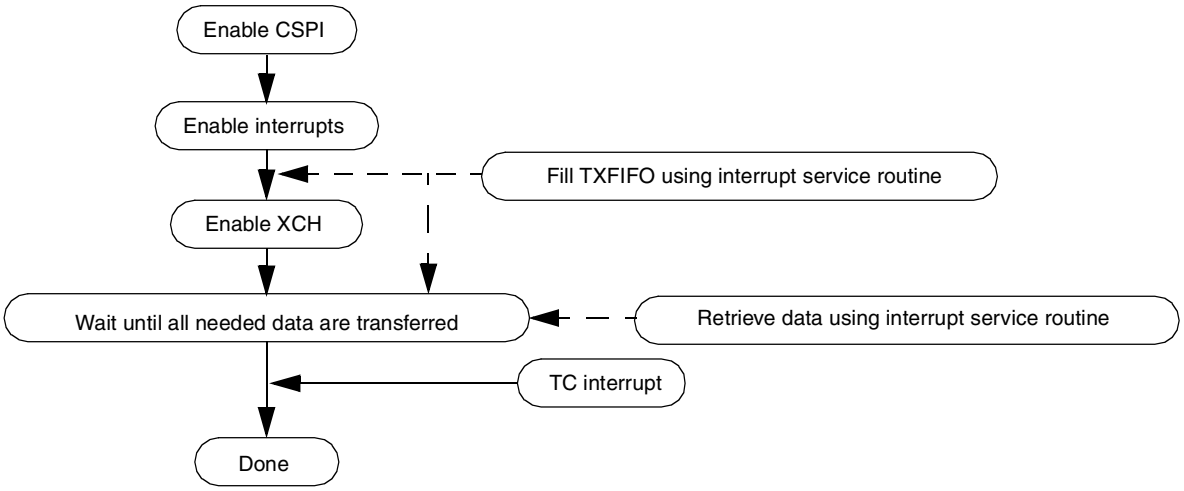


Figure 18-20. Program Sequence of SPI Burst Using Interrupt

18.4.4 DMA Control

DMA control provides another way to utilize the FIFOs in the CSPI module. Peripherals such as the CSPI which support DMA, use DMA request and acknowledge signals. Larger amounts of data can be transferred using DMA control, thereby reducing interrupts and CPU loading. When the appropriate conditions are matched, the module sends out a DMA request. The DMA deals with the following cases: TXFIFO empty, TXFIFO half, RXFIFO half, and RXFIFO full.

Figure 18-21 shows a program sequence of SPI bursts using the DMA.

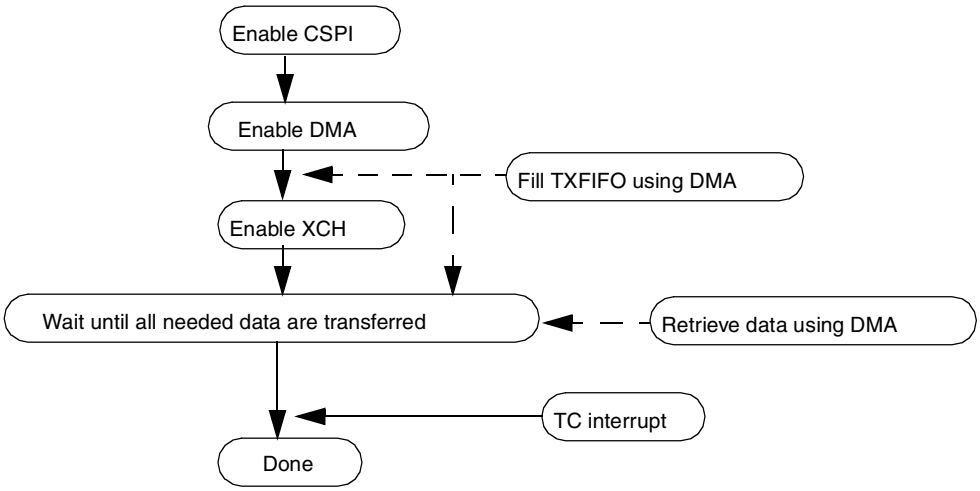


Figure 18-21. Program Sequence of SPI Burst Using DMA

18.5 Initialization/Application Information

This section provides initialization and application information for CSPI.

Figure 18-22 shows two flow charts for the master and slave mode of operations supported by the CSPI.

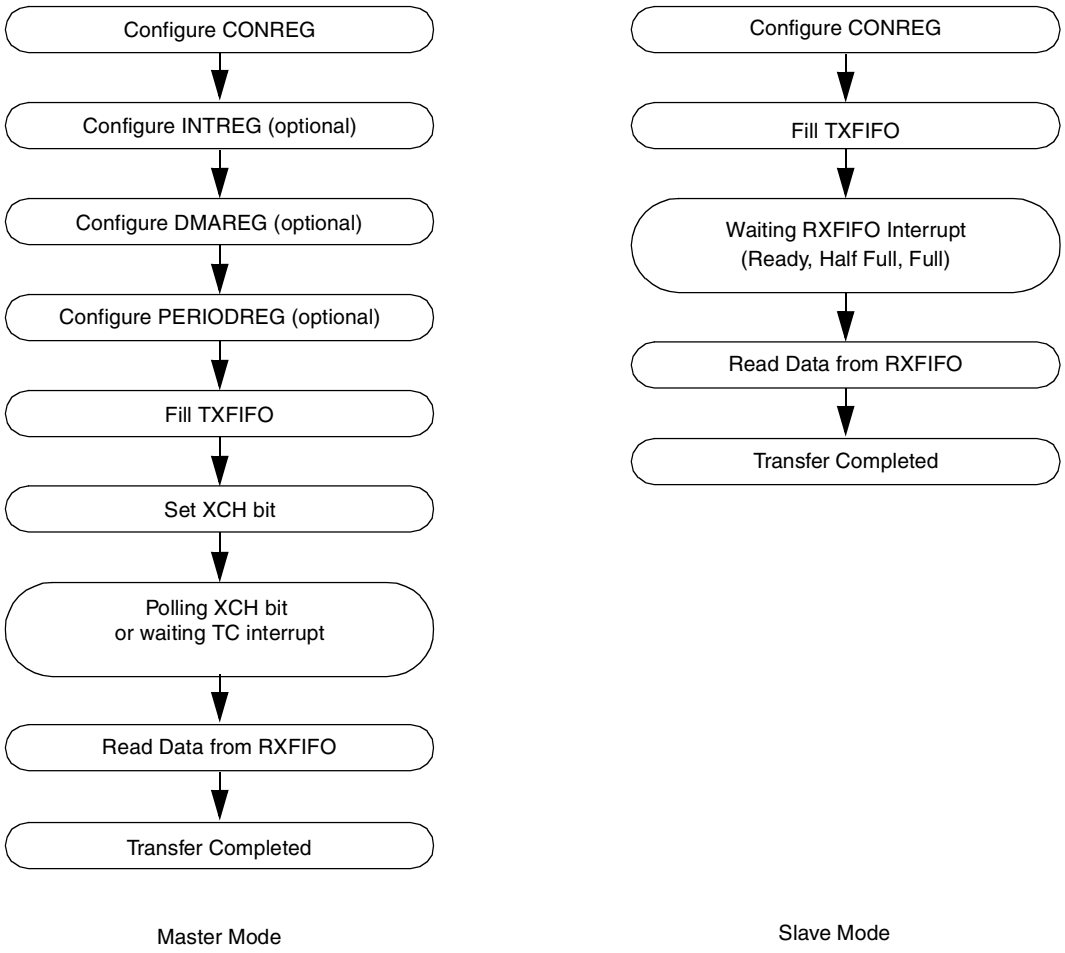


Figure 18-22. Flow Chart of CSPI Operation

Example 18-1 shows normal example code for CSPI operation using ARM instructions.

Example 18-1. CSPI Operation using ARM Instructions

```
LDR R0, =CSPI_BASE_ADDRESS          ; Load CSPI Base Address to R0
LDR R1, =0x01F00003                 ; Master Mode, 32-bit transaction
STR R1, [R0, #0x08]
LDR R1, =0x00000011                 ; Enable RXFIFO half and TXFIFO empty
STR R1, [R0, #0x0C]                 ; interrupt (Alternatively with DMA Mode)
LDR R1, =0x00000011                 ; Enable RXFIFO half and TXFIFO empty
STR R1, [R0, #0x10]                 ; DMA (Alternatively with interrupt)
LDR R5, =0x05                       ; R5 as number of words to be transferred.
LDR R1, =0x11111111                 ; R1 as increment to generate the data.
LDR R2, =0x12345678                 ; R2 load the data to be transferred.
```

```
Loop_00
        STR R2, [R2,#0x04]              ; Store data into TXFIFO.
        ADD R2, R2, R1                  ; Generating next data to be transferred.
        SUB R5, R5, #1                  ; Decrease the R5.
        CMP R5, #0x00                   ; Check R5 if it is zero.
        BNE Loop_00                     ; Loop until R5 is zero.

        LDR R1, =0x01F00007             ; set XCH bit to start transaction.
        STR R1, [R0, #0x08]
Loop_01
        LDR R1, [R0, #0x08]             ; check XCH bit if it is cleared.
        LDR R2, =0x00000004
        AND R1, R2, R1
        CMP R1, #0x00
        BEQ PASS_00                     ; if XCH bit is cleared then finish.
        B Loop_01                       ; if it isn't cleared then continue loop

        LDR R1, [R0, #0x00]             ; Read data from RXFIFO.
```

Chapter 19
Clock Amplifier (CAMP)

19.1 Overview

The Clock Amplifier converts a square wave/sinusoidal input of frequency range 8–40 MHz into a rail-to-rail square wave (camp supply voltage).

The input of the CAMP block is CKIH and the output is CAMP_OUT. The input to CAMP is internally AC coupled (in normal mode); no external coupling is required.

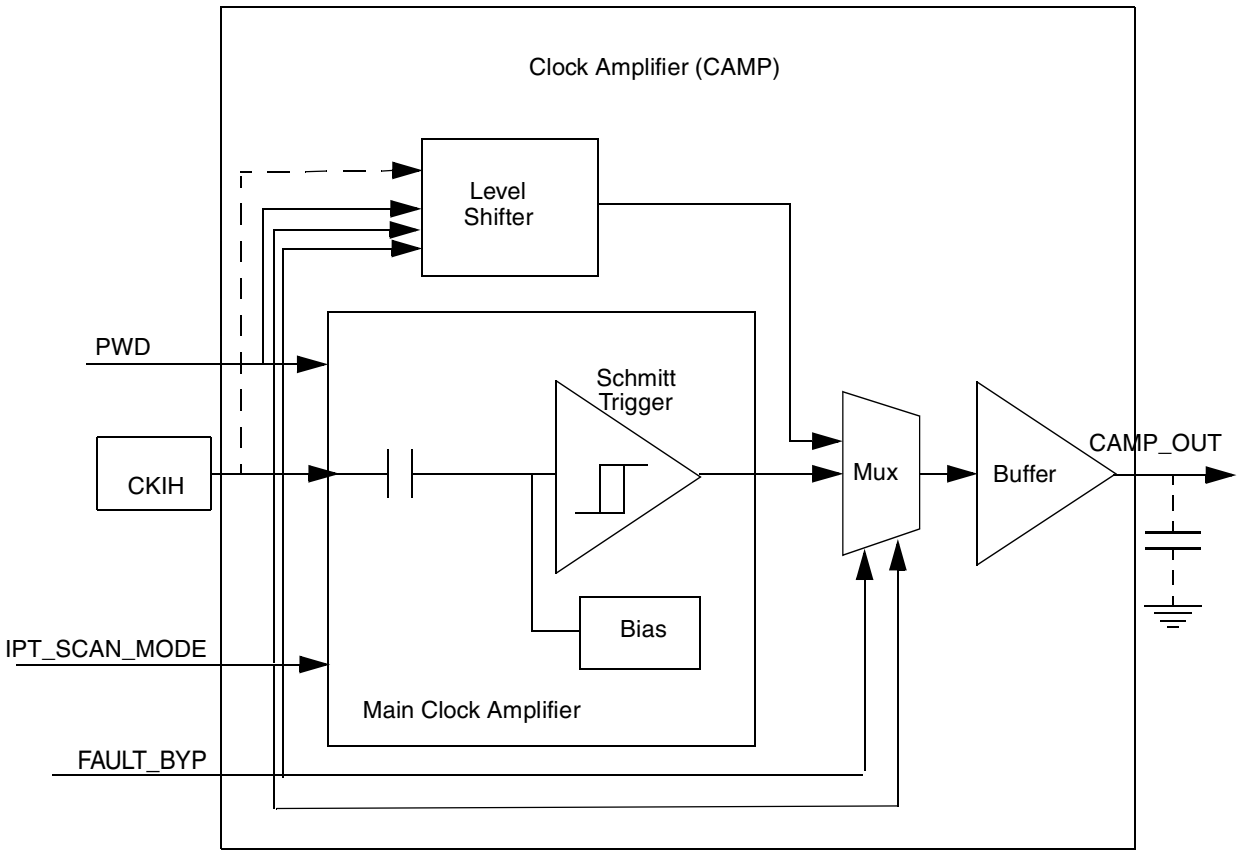


Figure 19-1. Clock Amplifier Block Diagram

19.1.1 Features

The CAMP includes the following features:

- Converts sinusoidal input to square wave.
- Can accept a square wave of amplitude greater than the supply voltage of the module.

19.1.2 Modes of Operation

The CAMP supports two modes of operation: normal and power down.

19.1.2.1 Normal Mode

In this mode of operation, the Clock Amplifier accepts a sinusodial/square wave as input and gives a rail to rail square wave output.

19.1.2.2 Power-Down Mode

In this mode the CAMP is disabled and put in the low-power state. CAMP_OUT remains at logic ’0’ level during power down.

19.2 External Signal Description

19.2.1 Overview

Table 19-1 shows the CAMP external signal properties.

Table 19-1. CAMP External Signal Properties

Name	Direction	Function
CKIH	Input	Input clock

Table 19-2 shows the CAMP interface signal properties.

Table 19-2. CAMP Interface Signal Properties

Name	Direction	Function
VDD	Input	Power supply
VSS	Input	Ground
PWD	Input	Power down signal
FAULT_BYP	Input	Test mode control signal
IPT_SCAN_MODE	Input	Scan mode signal
CAMP_OUT	Output	Output clock from CAMP

19.2.2 Detailed Signal Description

This section provides detailed signal descriptions.

19.2.2.1 CKIH—External clock input

The signal CKIH is the clock input signal to the CAMP. It can be a sinusoidal input with a minimum swing of 400 mV p-p or a square wave. It comes directly from the pad.

19.2.2.2 VDD—Power supply

The VDD is the power supply for the CAMP module. It supplies the main clock amplifier, buffer, and level shifter.

19.2.2.3 VSS—Ground

The VSS input is the ground for the CAMP.

19.2.2.4 IPT_SCAN_MODE—Scan Signal

This signal is active high. When this signal is asserted CAMP is forced to “Test Mode” irrespective of states of fault_byp or pwd signals.

19.2.2.5 PWD—Power Down Signal

This signal when asserted (and ipt_scan_mode = 0) puts the CAMP module in the low-power mode. This signal is active high.

19.2.2.6 FAULT_BYP—Control Signal for Test Mode

This signal when asserted puts CAMP in test mode. This signal is active high. When this signal is low (and ipt_scan_mode = 0) CAMP works in normal mode.

19.2.2.7 CAMP_OUT—Clock Output from CAMP

This signal is the output of the CAMP module.

19.2.3 Memory Map/Register Definition

The CAMP module obtains its control signals PWD and FAULT_BYP from registers residing in the CRM.

Chapter 20
Central Security Unit (CSU)

20.1 Overview

Security is an increasingly important feature of wireless mobile devices such as cell phones, ultra-portable computers and integrated media players. Instances of hackers and pirates breaking into portable devices and stealing private information and copyright content are becoming increasingly common. As such, security is a high priority for most high-tier architectures. To address the potential security risks and to provide an extensible platform for addressing future security needs, these designs incorporate a number of advanced hardware blocks and architectural features targeted at securing the platform. The Central Security Unit is one of them.

The Central Security Unit (CSU) enables software for setting comprehensive security policy within the platform and sharing secure information between various secure modules.

20.1.1 Features

The CSU has following security related features:

- System alarm routing policy
  - — Type of alarm triggered by the different system security violation indications
  - — Type of system event generated by each of the system alarms
  - — Security violation indication (SRTC) clearance (fuse driven)
- Setting emergency reset policy
- Execution mode access policy—Which peripheral resources can be accessed by what master privileges.
- Masters privilege policy—CSU helps those masters that cannot generate their own user/supervisor secure/non-secure signals to assert these signals.
- Special logic to prevent scan-in and scan-out of sensitive data

NOTE

For detailed information about the CSU module, contact your Freescale representative.

Chapter 21
Digital Phase Lock Loop (DPLL)

21.1 Introduction

The DPLL functional block diagram is shown in Figure 21-1. The external interface of the DPLL (I/O diagram) is shown in Figure 21-2.

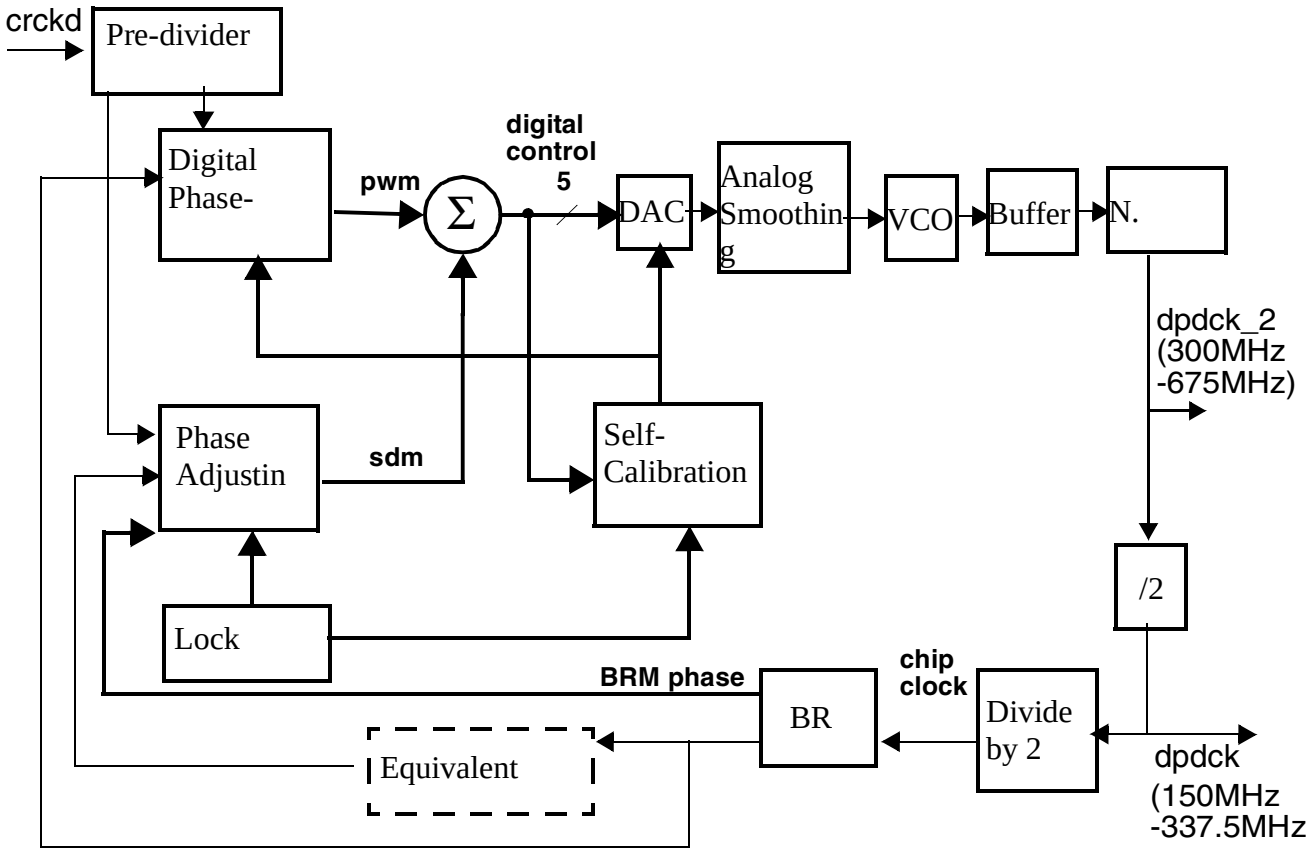


Figure 21-1. DPLL Block Diagram

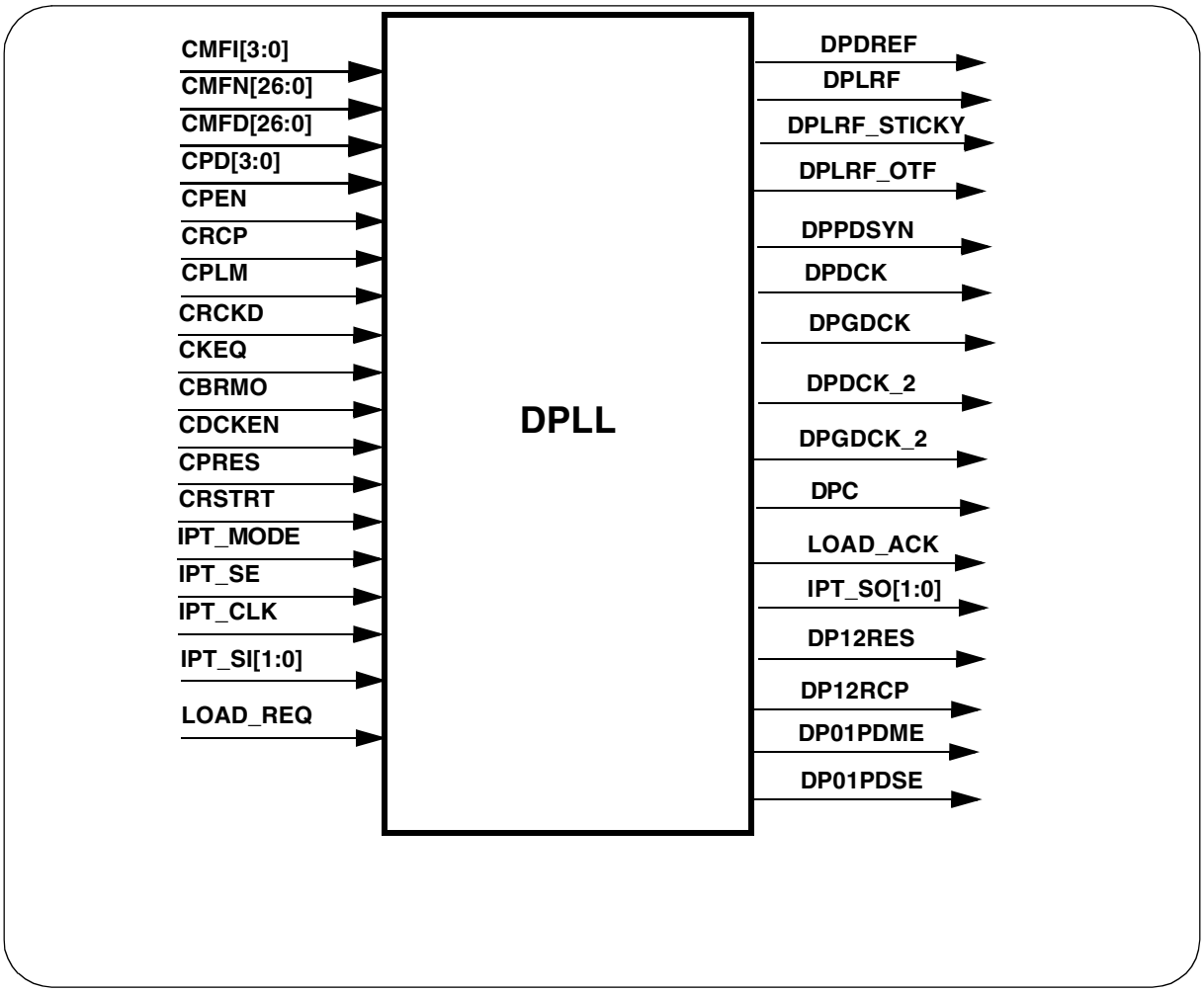


Figure 21-2. DPLL I/O Diagram

21.1.1 Overview

An on-chip Digital Phase Lock Loop (DPLL) provides clock generation in digital and mixed analog/digital chips designed for wireless communication and other applications. The DPLL produces a high-frequency chip clock with a low frequency and phase jitter.

21.1.2 Features

The DPLL includes the following features:

- A digital phase-frequency detection technique
- Fractional frequency multiplication method
- Digital control of an oscillator frequency
- Optional phase adjusting with digital phase loop filtering
- Areduced multiplication factor range

- Compensation of parameter variations by means of self-calibration

Digital implementation of frequency control and loop filtering functions allows the following new features:

- Eliminating an on-board loop filter capacitor, minimization of internal capacitor value, selection of frequency and phase/frequency operation modes
- An improved noise immunity, eliminating additional supply and ground pins
- A high frequency resolution with a reduced lock time
- Reduced sensitivity to parameter variations caused by temperature and process

The DPLL reference is an external system clock frequency. For wireless communication systems, the reference frequency is standardized. Most of the standard reference frequencies are located in the range of 10...100 MHz. The DPLL produces output clock (dpdck_2) at the frequency:

Eqn. 21-1

$$f_{dck\_2} = 4 \cdot f_{ref} \cdot \frac{MF_I + MF_N / MF_D}{PDF}$$

where f_{ref} is the reference frequency

MF_I is an integer part of a multiplication factor (MF)

MF_N and MF_D is the numerator and denominator of the MF fractional part

PDF is the predivision factor

Spectral purity of the DPLL output clock is characterized by phase and frequency jitter. Phase jitter is the magnitude of clock phase fluctuations relative to an ideal clock phase. Along with phase jitter, the output clock may be skewed relative to the reference clock. Frequency jitter is defined as magnitude of clock period fluctuation relative to an ideal clock period. Obviously, frequency jitter may be calculated as a difference of phase jitter values for adjacent clocks.

The DPLL communicates with the Clock Control Module (CCM)/DPLL-IP module. This CCM/DPLL-IP block contains a control register and provides an interface between the DPLL and various cores used in the SOC.

21.1.3 Modes of Operation

The DPLL operates in the following two modes: Frequency Only Lock mode and Phase Lock mode.

21.1.3.1 Frequency Only Lock Mode

For many stand-alone processors and asynchronous multiprocessor applications, only a frequency jitter value is important, and slow phase jitter and clock skew do not affect system performance. In such

systems, it is not necessary to adjust an output clock phase with a phase of input clock. A clock generation mode, for which the slow phase fluctuations are permissible, is named in this document as the Frequency Only Lock (FOL) mode. The DPLL operates in FOL mode when CPLM bit is made LOW.

21.1.3.2 Phase Lock Mode

A phase error can be important for synchronous applications and sampling A/D and D/A precision converters. The DPLL mode providing minimal phase jitter and skew elimination is mentioned as the Frequency and Phase Lock (FPL) mode. The DPLL operates in FPL mode when CPLM bit is HIGH & total multiplication factor is integer (ie. multiplication factor numerator is zero).

21.2 External Signal Description

This section provides and overview and list of properties for the external signals.

21.2.1 Overview

The DPLL I/O signals can be divided into the following categories:

- Reset signals
- Static control signals
- Clocks and other dynamic signals and
- Test mode signals (Freescale internal use only)
- BIST signals (Freescale internal use only)
- Supply signals

Table 21-1 gives the names and other properties of the DPLL I/O signals.

Table 21-1. DPLL I/O List

Signal	Dir.	Description	Reset State*	Default* Value
Reset signals				
cpres	I	Power up reset	1	1
cpen	I	DPLL enable	0	0
crstrt	I	Restart	N/A	N/A
Static control signals				
cmfi	I	Multiplication factor integer part	N/A	N/A
cmfd	I	Multiplication factor fractional part denominator	N/A	N/A
cpd	I	Predivision factor	N/A	N/A
crcp	I	Reference clock polarity bit	N/A	N/A
cplm	I	Phase lock mode bit (0 - FOL, 1 - FPL)	N/A	N/A
cbrmo	I	BRM order bit (0 - first, 1 - second)	N/A	N/A

Table 21-1. DPLL I/O List (continued)

Signal	Dir.	Description	Reset State*	Default* Value
Clocks and other dynamic signals				
cmfn	I	Multiplication factor fractional part numerator	N/A	N/A
crckd	I	Reference clock frequency	N/A	N/A
ckeq	I	Clock delayed by equivalent delay	N/A	N/A
cdcken	I	Gated output clock enable signal	N/A	N/A
load_req	I	Load request for on the fly change of cmfn	N/A	N/A
dplrf	O	Lock ready flag	1'b0	N/A
dplrf_sticky	O	Sticky lock ready flag	1'b0	N/A
dplrf_otf	O	On the fly lock ready flag	1'b0	N/A
dpdref	O	Divided reference clock	1'b0	N/A
dppdsyn	O	Post-divider synchronization signal	1'b1	N/A
dpdck	O	Output clock divide by 2	1'b0	N/A
dpdck_2	O	Output clock	1'b0	N/A
dpgdck	O	Gated output clock divide by 2	1'b0	N/A
dpgdck_2	O	Gated output clock	1'b0	N/A
dpc	O	Clock before the equivalent delay	1'b0	N/A
load_ack	O	On the fly cmfn change acknowledgement	1'b0	N/A

21.2.2 Detailed Signal Descriptions

This section provides detailed signal descriptions.

21.2.2.1 Reset Signals: cpres, cpen, and crstrt

There are three reset signals for the DPLL: the power-up reset CPRES, DPLL enable CPEN, and restart CRSTRT signals.

The DPLL is operating when CPRES and CRSTRT are '0' and CPEN is '1'; otherwise it is disabled.

21.2.2.1.1 cpres

This is the power up reset signal. The DPLL is reset when cpres = 1'b1

21.2.2.1.2 cpen

This is the DPLL enable signal. The DPLL is reset when cpen = 1’b0

21.2.2.1.3 crstrt

This is the DPLL restart signal. When crstrt = 1’b1, all DPLL module are reset, excluding the current reference.

The DPLL states are given in Table 21-1.

Table 21-2. DPLL Reset States

CPRES	CPEN	CRSTRT	State
1	X	X	All DPLL modules are reset
X	0	X	
0	1	1	All DPLL modules are reset excluding the current reference
0	1	0	DPLL is operating

After the CPRES signal has been set, the entire DPLL module is immediately reset. When the DPLL is powered on at chip start up, CPRES must be asserted (ie. made "1") and then de-asserted (ie. made "0"). This ensures that all the internal registers are set/reset at their desired values. During normal operation, the DPLL can be reset using CPEN or CRSTRT signals. Reset caused by the CPEN and CRSTRT signals creates a delay of three/four output clock (dpdck) cycles for dpgdck and three/four output clock (dpdck_2) cycles for dpgdck_2. It stops the output clock series synchronously as shown in Figure 21-3.

NOTE

The DPLL must be reset using "cpen" or "cpres" signal instead of restarting by "crstrt" signal when it is desired to be locked according to new DPLL settings (modes).

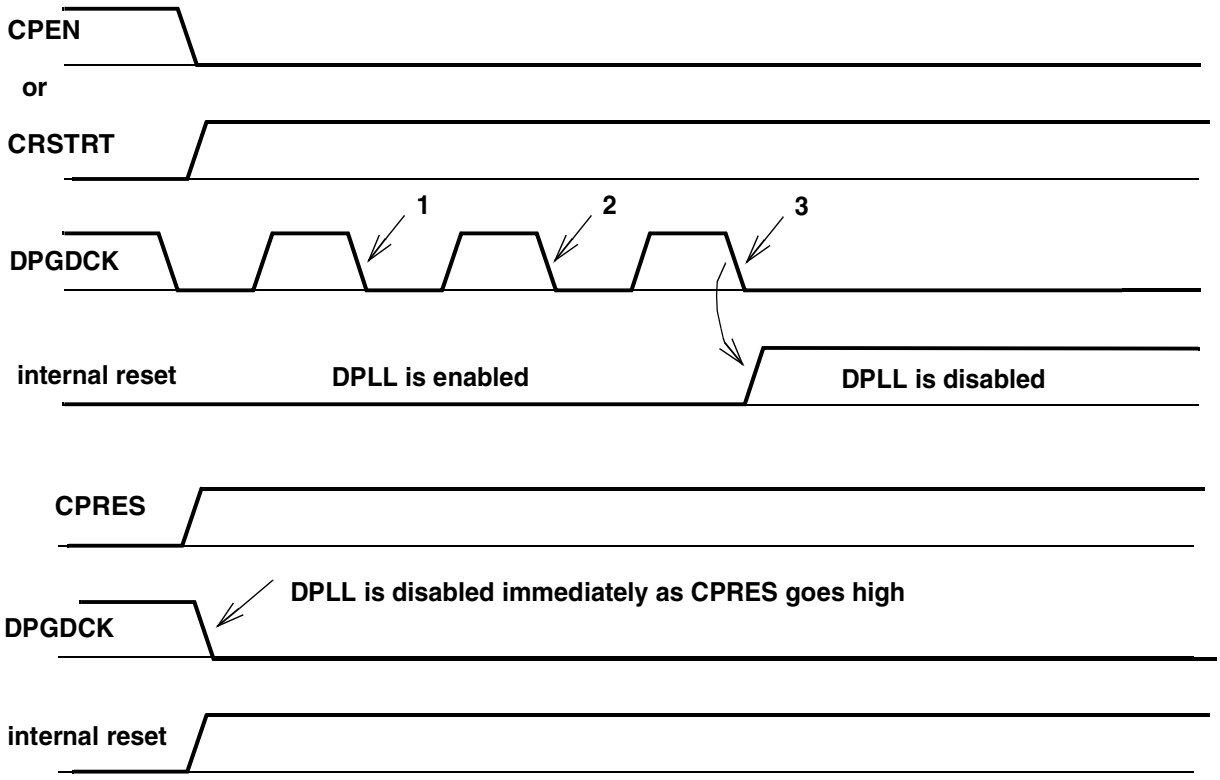


Figure 21-3. Timing diagram for DPLL stop

The DPLL starts after three reference clock periods as shown in Figure 21-4. This is provided by a synchronizer. The VCO is enabled simultaneously by enabling all the DPLL logic, but oscillation starts after settling the internal current reference. If the DPLL was disabled using only the CRSTRT signal (**partial reset**), the lock-in time is reduced by 128 divided reference clocks.

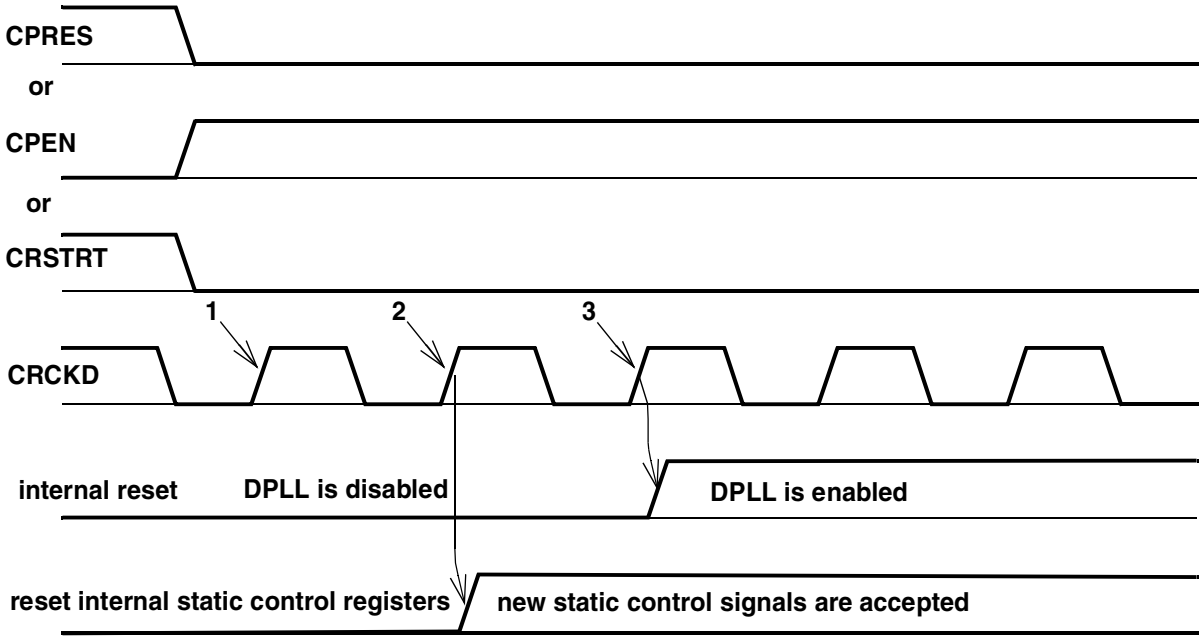


Figure 21-4. Timing diagram for DPLL start

21.2.2.2 Static Control Signals: cmfi, cmfd, cpd, crcp, cplm, cbrmo

The static control signals are discussed in the following subsections.

21.2.2.2.1 cmfi

Multiplication Factor Integer Part bits (cmfi[3:0]).

The multiplication factor integer part. cmfi[3:0] bits should be in a range from 5 to 15. If it is less than 5, the DPLL accepts 5.

21.2.2.2.2 cmfd

Multiplication Factor Fractional Part Denominator bits (cmfd[26:0])

The cmfd[26:0] bits, in a 2’s compliment format, give the denominator (bit 26 is always zero). Internal to DPLL, the cmfd[26:0] number should be in a range from 1 to 67,108,863; otherwise, the output clock frequency will differ from the desired frequency. The cmfd bits are ignored if the MF numerator (cmfn) is zero. Internal to the DPLL, a “1” is added to the CMFD number supplied by the user. For user, the CMFD range will be from 0 to 67108862.

21.2.2.2.3 cpd

Predivision Factor Minus One (cpd[3:0])

The cpd[3:0] bits give the predivision factor minus 1. The cpd[3:0] number should be in a range from 0 to 15.

21.2.2.2.4 crcp

Reference Clock Polarity bit.

If the crcp bit is cleared, the chip clock (dp08c0) is adjusted with a positive edge of the reference clock. If the bit is set, the chip clock is adjusted with its negative edge.

21.2.2.2.5 cplm

Phase Lock Mode bit.

The DPLL operates in the Frequency Only Lock (FOL) mode, when the cplm bit is cleared (1’b0), and in Frequency and Phase Lock mode (FPL), when the bit is set (1’b1). The FPL mode can be used for both an integer and fractional multiplication factor, but phase skew elimination is accomplished only for the integer MF.

21.2.2.2.6 cbrmo

BRM order bit

When the cbrmo bit is cleared (cbrmo=1’b0), the BRM has first order, otherwise the BRM order is 2. The first-order BRM should be used only if the MF fractional part denominator is less than 8. In other cases, the BRM order should be 2. This bit is ignored if the MF numerator is zero.

21.2.2.3 Clocks and Other Dynamic Signals

The clocks and dynamic siglans are as follows: cmfn, crckd, cdcken, load_req, dplrf, dpdref, dppdsyn, dpdck, dpdck_2, dpgdck, dpgdck_2, dpc, load_ack, dplrf_sticky, and dplrf_otf. They are discussed in the following subsections.

21.2.2.3.1 cmfn

Multiplication Factor Fractional Part Numerator bits (cmfn[26:0])

The cmfn[26:0] bits, in a 2’s compliment format, give the numerator. The cmfn[26:0] bits are the only bits in the DPLL that can be changed after the DPLL was locked without resetting the DPLL (on the fly MFN change). The cmfn[26:0] number should be in a range from —67108862 to 67108862. Bit 26 is the sign bit.If the absolute value of the numerator is larger than denominator (i.e. |numerator| > denominator), the output clock frequency will differ from the desired frequency. If the numerator is zero, the circuitry for fractional division is disabled to save power.

21.2.2.3.2 crckd

reference clock

The reference clock frequency can be in a range from 10 to 100MHz. The reference clock duty cycle should be 50 ± 20%.

21.2.2.3.3 ckeq

The CKEQ signal is fedback to DPLL and is a delayed copy of DPC after equivalent delay.

21.2.2.3.4 cdcken

Gated output clock enable signal.When cdcken = 1’b1, dpgdck and dpgdck_2 are enabled.

The dpgdck and dpgdck_2(gated output clocks) output signal are obtained from the output clock (dpdck and dpdck_2 respectively) by gating with the cdcken input signal. Because cdcken is synchronized in the DPLL, no spikes or shorted phases take place. Figure 21-5 shows the gating timing diagrams.

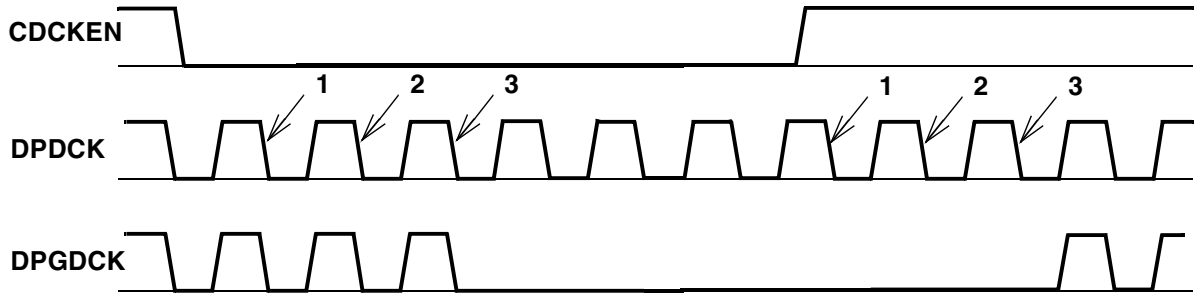


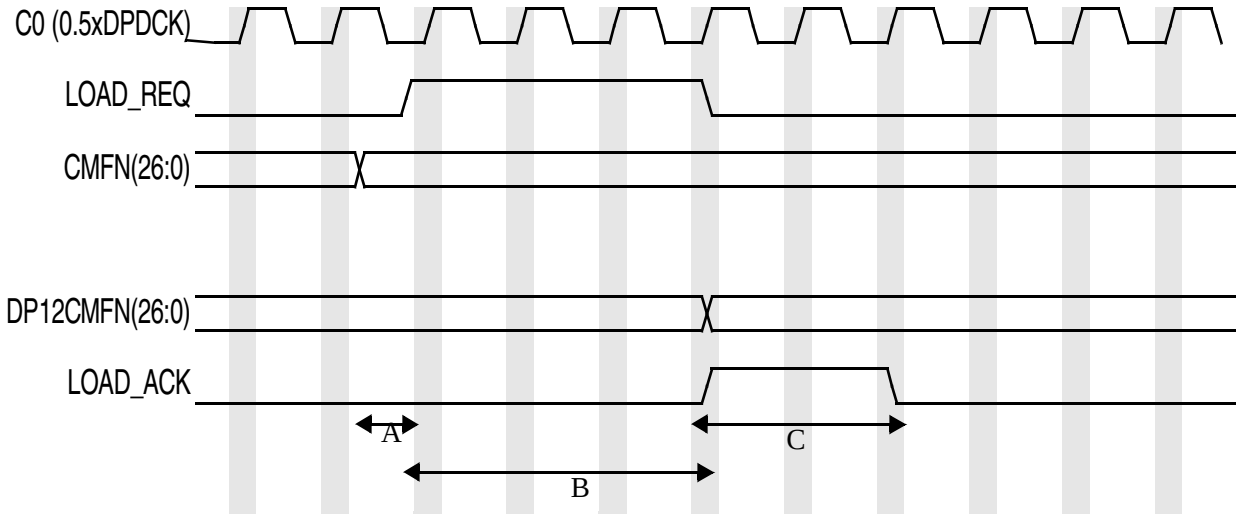
Figure 21-5. Clock Gating Timing Diagram

Both dpgdck and dpgdck_2 start/stop synchronously with delay of three/four cycles of dpdck and dpdck_2 respectively.

21.2.2.3.5 load_req

Load request for on the fly change of cmfn.

When on the fly cmfn value change is required, load_req should be made high. Figure 21-6 provides the timing diagram for this.



1 DP12CMFN represents the latched value of CMFN.
2 Load Request should be asserted after CMFN change

Figure 21-6. Timing diagram for CMFN

- A —The time difference between LOAD_REQ and the new value of CMFN should be greater than or equal to zero.
- B—The duration from posedge LOAD_REQ to the actual loading of the CMFN new value to the PLL is atleast three clock cycles of dp08c0 clock which has half the frequency of DPLL output clocks, dpdck/dpgdck.Thus, in terms of dpdck/dpgdck, the minimum duration of LOAD_REQ is atleast six clock cycles of dpdck/dpgdck.
- C—LOAD_ACK signal duration is two clocks.

Minimum frequency resolution will be achieved with minimum CMFN(ie equal to 1 or -1) and maximum CMFD (ie equal to 67108863). For dpdck_2 clock, the minimum frequency resolution will be twice that of dpdck clcok.

21.2.2.3.6 dplrf

Lock ready flag.

This output signal, when set, indicates that the DPLL is in lock.

21.2.2.3.7 dpdref

Divided reference clock.

This is the divided reference clock signal. Its frequency is given as

Eqn. 21-2

$$f_{dpdref} = \frac{f_{ref}}{PDF}$$

21.2.2.3.8 dppdsyn

Post divider synchronization. The dppdsyn signal is the DPLL output intended for synchronization of a post-divider that can follow the DPLL. Synchronization ensures a constant and minimal phase offset between the reference and post-divider output clocks. The dppdsyn signal is used only for an integer MF (Figure 21-7). When MF is non-integer, dppdsyn is zero.

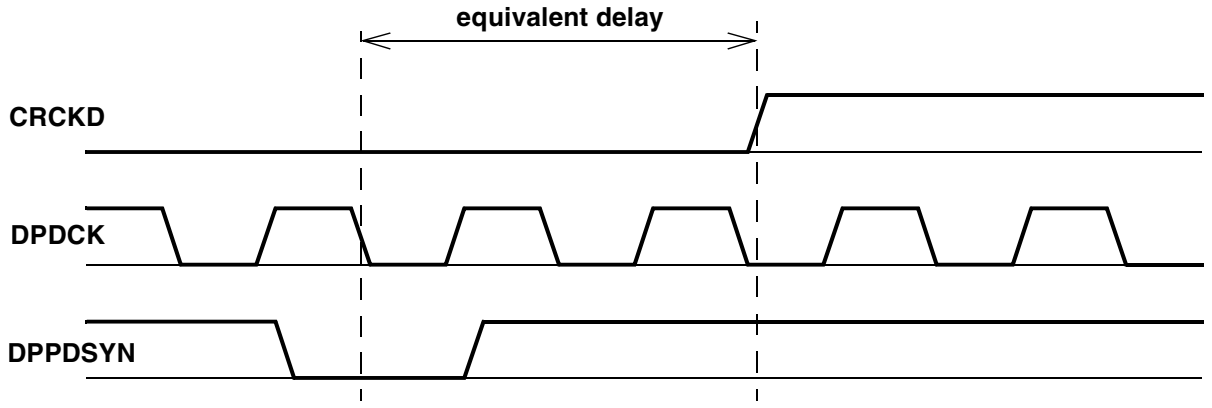


Figure 21-7. Timing diagram for DPPDSYN signal

21.2.2.3.9 dpdck

Output clock divide by 2 (ungated)

dpdck clock is the DPLL output clock divided by 2 and has 50 ± 1.5% duty cycle. This signal is zero during reset.

21.2.2.3.10 dpdck_2

Output clock (ungated)

dpdck_2 clock is the DPLL output clock and has 50% ± 1.5% duty cycle. This signal is zero during reset.

21.2.2.3.11 dpgdck

Gated output clock divide by 2. The frequency will be half the frequency of gated output clock.This signal is zero during reset.Pl. Refer to Figure 21-6 on page 21-11 for more details.

21.2.2.3.12 dpgdck_2

Gated output clock. This signal is zero during reset.

21.2.2.3.13 dpc

Clock before the equivalent delay. The output of the equivalent delay is CKEQ signal.

21.2.2.3.14 load_ack

On the fly CMFN change acknowledgement signal. This is an output of the DPLL. When the changed value of CMFN is registered in the DPLL, this signal goes high.Pl. Refer toFigure 21-7 on page 21-12 for more details.

21.2.2.3.15 dplrf_sticky

This is the sticky bit implementation of dplrf signal. In FPL mode, dplrf goes high when DPLL meets its lock criteria and goes low when DPLL looses lock. The dplrf_sticky will remain high even if dplrf goes low.

21.2.2.3.16 dplrf_otf

On the fly lock ready flag. This signal goes high after TBD number of divided reference clock cycles after load_ack goes high.

21.2.2.4 Supply Pins

The supply pins are as follows:

- avdd—DPLL analog section power supply.
- avss—DPLL analog section ground supply.
- dvdd—DPLL digital section power supply.
- dvss—DPLL digital section ground supply.
- sub—P substrate connection for guard ring only.

Chapter 22
DPLL-IP Interface (DPLL-IP)

22.1 Overview

This module serves as an interface to the DPLL module. It generates the control signals required for the DPLL operations. In other MX ICs this used to be part of the SRC (System Reset Controller) module. The separation of the DPLL from the SRC allows the flexibility for DPLL to be controlled by any of the processors that have ownership of it.

22.1.1 Feature Description

Key features of this module include:

- Maintains control/operation registers for DPLLs
- Uses a 32-bit IP bus interface, with all its registers byte, half word, and word accessible
- Provides controlled phase modulation of clocks to reduce receiver desensitization using desense circuit for DPLLs
- Generates DPLL-Enable signal (**dpllip_cpen**) depending upon both the **dpll_en_dpllip** signal (coming from SRC) as well as the internal register UPEN bit
- Selects the reference clock for DPLL (also goes to SRC) from **clock0, clock1, clock2,** and **clock3** depending upon the **ref_clk_sel** bits (bit-8 and bit-9) of the DP_CTL register
- If auto restart is enabled (**AREN** bit is ‘1’), a restart sequence is issued automatically whenever one of the DPLL registers (CTL, OP, MFD or HFS_OP, HFS_MFD) is written. Otherwise, software needs to set the restart bit manually.
- Allows SJC to control pll-enable, restart and MFI values through inputs from SJC
- Provides multiple options for Controlling DPLL On/Off independently by software as well as by DSM (Deep Sleep Mode)

22.1.2 Modes of Operation

The DPLL interface supports the following modes of operation:

- Normal Mode—supports low-frequency settings.
- HFS Mode—supports high-frequency settings
- Desense Mode—provides controlled phase modulation of clocks to reduce receiver desensitization using desense circuit for DPLL.

22.2 Memory Map/Register Definition

22.2.1 Register Summary

The register summary lists all registers of the module. The absolute address of each register is given, and the value of each bit for reads and writes is given using the conventions in the legend.

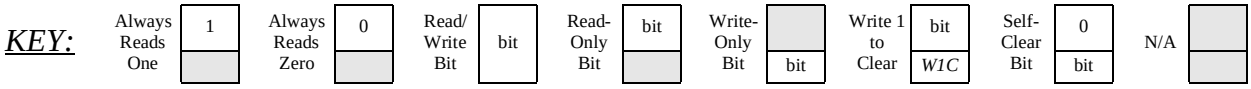


Figure 22-1. Register Figure Key

Table 22-1. Interface Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP_CTL (0xBASE +0x000)	R	0															
	W																
	R			MUL_CTRL	DPDCK0_2_EN	ADE	REF_CLK_DIV	REF_CLK_ SELREF_CLK _SEL[1:0]		HFSM	PRE	UPEN	RST	RCP	PLM	BRMO	LRF
	W																
DP_CON FIG (0xBASE +0x004)	R	0															
	W																
	R	0												BIST _CE	SJC _CE	AREN	LDREQ
	W																
DP_OP (0xBASE +0x008)	R	0															
	W																
	R	0								MFI[3:0]				PDF[3:0]			
	W																
DP_MFD (0xBASE +0x00C)	R	0					MFD[26:16]										
	W																
	R	MFD[15:0]															
	W																
DP_MFN (0xBASE +0x010)	R	0					MFN[26:16]										
	W																
	R	MFN[15:0]															
	W																

Table 22-1. Interface Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP_MFN MINUS (0xBASE +0x014)	R	0					MFNMINUS[26:16]										
	W																
	R	MFNMINUS[15:0]															
	W																
DP_MFN PLUS (0xBASE +0x018)	R	0					MFNPLUS[26:16]										
	W																
	R	MFNPLUS[15:0]															
	W																
DP_HFS _OP (0xBASE +0x01C)	R	0															
	W																
	R	0								HFS_MFI[3:0]				HFS_PDF[3:0]			
	W																
DP_HFS _MFD (0xBASE +0x020)	R	0					HFS_MFD[26:16]										
	W																
	R	HFS_MFD[15:0]															
	W																
DP_HFS _MFN (0xBASE +0x024)	R	0					HFS_MFN[26:16]										
	W																
	R	HFS_MFN[15:0]															
	W																
DP_MFN _TOGC (0xBASE +0x028)	R	0														TOG_ DIS	TOG_E N
	W																
	R	TOG_CNT[15:0]															
	W																
DP_DES TAT (0xBASE +0x02C)	R	TOG_ SEL	0				TOG_MFN[26:16]										
	W																
	R	TOG_MFN[15:0]															
	W																

22.2.2 Detailed Register Descriptions

The figures and associated text in the subsequent sections provide detailed descriptions of the clock control registers. The following definitions serve as a key for these figures:

- **Grey bit**: unimplemented bit. Always reads as zero; Writing has no effect.
- **TYPE**: Type of register bit. Defines register bit's behavior. Possible values:
  - **r**: read only. Writing this bit has no effect.
  - **w**: write only.
  - **rw**: Standard read/write bit. Only software can change bit's value (other than a hardware reset).
  - **rwm**: A read/write bit that may be modified by a hardware in some fashion other than reset.
  - **w1c**: A status bit that can be read, and it cleared by writing a logic 1.
  - **slfclr**: Self clearing bit. Writing a one has some effect on module, but it always reads as 0.
- **RESET**: Gives the reset value of the bit. Possible values:
  - **0**: Will reset to a logic 0
  - **1**: Will reset to a logic 1
  - **?**: The reset state is unknown.
  - **u**: Unaffected by reset

22.2.2.1 DPLL Control Register (DP_CTL)

Figure 22-2 shows the DP_CTL register diagram and the field descriptions are found in Table 22-2.

0xBASE+0x000 (DP_CTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	DPDCK02_EN	0	REF_CLK_DIV	REF_CLK_SEL[1:0]		HFSM	PRE	UPEN	RST	RCP	PLM	BRMO	LRF
W			MUL_CTRL													
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-2. DPLL Control Register Diagram

Table 22-2. DP_CTL Field Descriptions

Field	Description
31–14	Reserved
13 MUL_CTRL	Multiple Control. This is a self-clearing bit. and is cleared on the next posedge of PLL reference clock. Setting this bit deasserts the enable signal to the DPLL. Using this bit instead of UPEN to disable the DPLL gives DSM the control to independently switch on the DPLL. Details on the usage of this bit are given in Section 22.3.4, "Multiple Options for DPLL Control" on page 22-19. 0 No Effect. 1 DPLL disabled.
12 DPDCK0_2_EN	dpdck0_2 Enable. This bit enables the PLL output before the divide-by-2 flip-flop. It enables clock dpdck0_2 whose output is double the frequency of dpdck/dpgdck clock. 0 dpdck0_2 disable 1 dpdck0_2 enable
11	Reserved
10 REF_CLK_DIV	Ref Clk Division factor. Determines if the selected reference clock input is divided by a factor of two or one. Reset value of this bit is controlled by the init_dp_ctl_dpllip[10] bit. 0 divide by 1 1 divide by 2
9–8 REF_CLK_SEL[1:0]	Reference Clock Select:- This field selects one of the four clocks for reference clock for DPLL as listed next. Reset value of this bit is determined by the setting of init_dp_ctl_dpllip[9:8]. For all three DPLL-IP's: - clk 0 is not connected - clk 1 is not connected. - clk 2 is COSC (internal oscillator) - clk 3 is FPM. 00 clock0 is selected 01 clock1 is selected 10 clock2 is selected 11 clock3 is selected
7 HFSM	HFS-Mode Status bit. This read-only bit is a status bit that indicates which frequency mode of operation is active. In HFS mode all the shadow registers (DP_HFS_OP, DP_HFS_MFD, DP_HFS_MFN) are used whereas in LFS mode the normal registers are used. The reset value of this bit it ‘0’. 0 LFS mode, normal registers are read. 1 HFS mode is active
6 PRE	Power Up Reset. Asserts a hard reset to the DPLL. 0 reset cleared. 1 reset asserted
5 UPEN	PLL Enable. Enables DPLL operation. 0 PLL disabled. 1 DPLL enabled.
4 RST	Restart. Restarts the DPLL. It should not be used as a status bit. 0 DPLL not in restart. 1 DPLL restarted.
3 RCP	Reference Clock Polarity. Selects which edge the chip clock is adjusted to. 0 positive edge of reference clock. 1 negative edge of reference clock.

Table 22-2. DP_CTL Field Descriptions (continued)

Field	Description
2 PLM	Phase Lock Mode. Selects if phase is to be considered (in addition to frequency) during lock-in. 0 DPLL set to frequency only lock mode. 1 DPLL in frequency and phase lock mode.
1 BRMO	BRM Order Bit. Binary Rate Modulator order. 0 BRM in first order. 1 BRM in second order.
0 LRF	Lock Ready Flag. This bit indicates when the DPLL is in lock. This is a sticky bit. The reset value of this bit is '0'. The following events will break the already established lock. • DP_CTL, DP_OP, DP_MFD, DP_HFS_OP, or DP_HFS_MFD is written • Hard reset • pll_lvs toggle • DPLL enable 0 DPLL not locked. 1 DPLL locked.

22.2.2.2 DPLL Configuration Register (DP_CONFIG)

Figure 22-3 shows the DP_CONFIG register diagram, and Table 22-3 provides the field descriptions.

0xBASE+0x004 (DP_CONFIG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	AREN	LDREG
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 22-3. DPLL Configuration Register Diagram

Table 22-3. DP_CONFIG Field Descriptions

Field	Description
31–2	Reserved
1 AREN	Auto Restart Enable. If auto restart is enabled, then a restart sequence is issued automatically whenever a write is performed to one of the DPLL registers (DP_CTL, DP_OP, DP_MFD) in normal mode. In HFS mode, the same is true if a write is performed to one of the HFS mode registers (DP_CTL, DP_HFS_OP, or DP_HFS_MFD). Otherwise, software will need to set the restart bit manually. 0 Auto-restart is disabled 1 Auto-restart is enabled (Default Value of this bit is '1')
0 LDREQ	Load Request. Setting this bit notifies the DPLL that the numerator factor of the fractional part (MFN) has changed. This bit is cleared by an acknowledge from the DPLL. Writing a '0' to this bit has no effect. 0 DPLL has finished updating MFN. 1 request the DPLL update MFN. (Default Value of this bit is '0')

NOTE

Writing to the DP_CONFIG register does not generate an automatic 'restart'.

22.2.2.3 DPLL Operation Register (DP_OP)

Figure 22-4 shows the DP_OP register diagram and Table 22-4 provides the field descriptions. The MFI and PDF fields represent the decimal value of MFI and PDF respectively, which is output to the DPLL module by the DPLL-IP.

0xBASE+0x008 (DP_OP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	MFI				PDF			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-4. DPLL Operation Register Diagram

Table 22-4. DP_OP Description

Field	Description
31–8	Reserved
7–4 MFI	Multiplication Factor Integer. This field defines the integer portion of the multiplication factor (MFI) If MFI is written with a value of less than 5,then MFI defaults to 5. 0000 5 ... 0101 5 0110 6 ... 1111 15
3–0 PDF	Predivision Factor (PDF). The value written to PDF[3:0] is equal to the Predivider Factor minus one. 0000 0 0001 1 0010 2 ... 1110 14 1111 15

Registers DP_OP, DP_MFN, and DP_MFD calculate the output frequency by the formula shown in Equation 22-1:

Eqn. 22-1

$$f_{dck} = 4 \cdot f_{ref} \cdot \left[\frac{MF}{PDF + 1}\right] \qquad \text{Where: } MF = MF_I + \frac{MF_N}{MF_D + 1}$$

22.2.2.4 DPLL MFD Register (DP_MFD)

Figure 22-5 shows the DP_MFD register diagram and Table 22-5 provides the field descriptions. The MFD is the decimal value of MFD[26:0] which is provided to DPLL by DPLL-IP module. Inside the DPLL, MFD is incremented by 1.

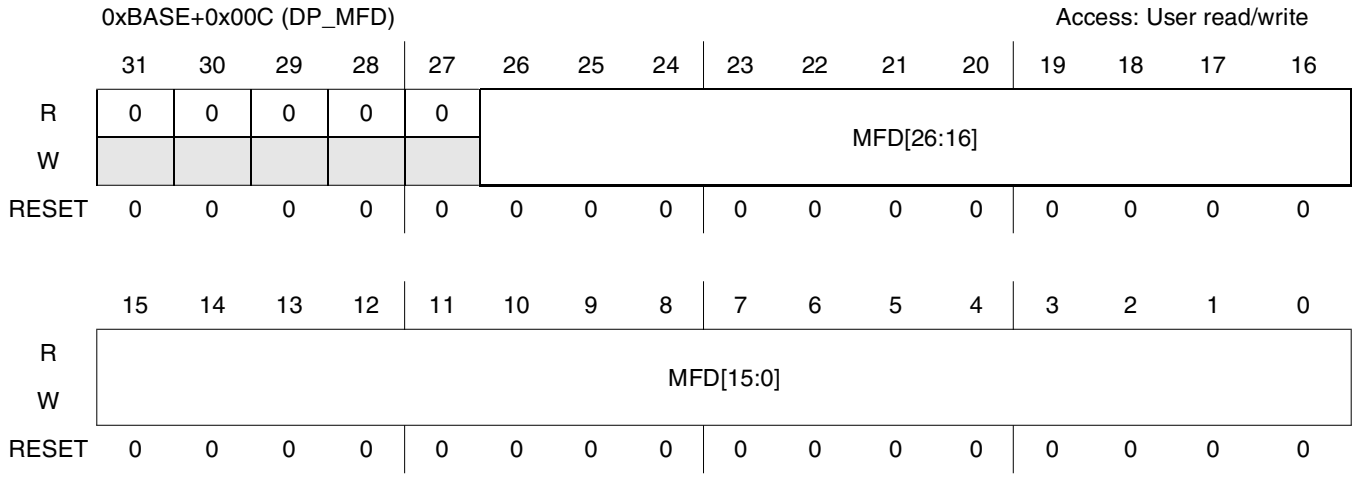


Figure 22-5. DPLL MFD Register Diagram

Table 22-5. DP_MFD Field Description

Field	Description
31–27	Reserved
26–0 MFD[26:0]	Multiplication Factor Denominator Bits. The bits in the MFD[26:0] field contain the denominator of the fractional part in a 2's compliment format. The range of the MFD is from 0 to 67108863. Settings outside of this range cause the output clock frequency to differ from the desired frequency. **Note:** Bit 26 is always 0. Settings (Hex) 000000 0 000001 1 000002 2 ... 3FFFFE 67108862 3FFFFF 67108863

22.2.2.5 DPLL MFNnnn Registers (DP_MFN, DP_MFNMINUS, and DP_MFNPLUS)

Figure 22-6 shows the DP_MFDnnn register diagram and Table 22-6 provides the field descriptions. MFNnn denotes MFN, MFNPLUS and MFNMINUS registers. The MFNPLUS and MFNMINUS

registers are part of the Desense logic. The MFN field contains the decimal value of MFN[26:0], which is output to the DPLL module by DPLL-IP.

0xBASE+0x010 (DP_MFN) Access: User read/write
0xBASE+0x014 (DP_MFNMINUS)
0xBASE+0x018 (DP_MFNPLUS)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	MFN[26:16]										
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MFN[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-6. DPLL MFNnnn Registers Diagram

Table 22-6. DP_MFNnnn Field Description

Field	Description
26–0 MFNxxx	Multiplication Factor Numerator Bits (nominal, “MINUS”, or “PLUS”)— The MFNxxx[26:0] bits, in a 2’s compliment format, give the numerator of the fractional part. The MFN range is from -67108864d to 67108863d. If the absolute value of the MFN is larger than MFD, the output clock frequency will differ from the desired frequency. The default value of the MFNMINUS and MFNPLUS registers is 0. Settings (Hex) 4000000 -67108864 4000001 -67108863 4000002 -67108862 ... FFFFFFF -1 0000000 0 (default) 0000001 1 ... 3FFFFFE 67108862 3FFFFFF 67108863

22.2.2.6 DPLL _High Freq. Support, Operation Register (DP_HFS_OP)

Figure 22-7 shows the DP_HFS_OP register diagram and Table 22-7 provides the field descriptions. The DP_HFS_OP register is used to program the decimal values for the multiplication factor integer

(HFS_MFI) and the pre-division factor (HFS_PDF) respectively, which are output to the DPLL module by the DPLL-IP module.

0xBASE+0x01C (DP_HFS_OP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	HFS_MFI				HFS_PDF			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-7. DPLL _High Freq. Support, Operation Register

Table 22-7. DP_HFS_OP Field Descriptions

Field	Description
31–8	Reserved
7–4 HFS_MFI	HFS mode, Multiplication Factor. This field controls the Integer part of the multiplication factor (MFI). If HFS_MFI is written with a value less than 5,then MFI defaults to 5. 0000 5 ... 0101 5 0110 6 ... 1111 15
3–0 HFS_PDF	HFS mode, Pre-division Factor (HFS_PDF). The value written to HFS_PDF[3:0] is equal to the Pre-Divider Factor minus one. 0000 0 0001 1 0010 2 ... 1110 14 1111 15

22.2.2.7 DPLL High Freq. Suppport MFD Register (DP_HFS_MFD)

Figure 22-5 shows the DP_MFD register diagram and Table 22-5 provides the field descriptions. The MFD is the decimal value of MFD[26:0] which is provided to DPLL by DPLL-IP module.

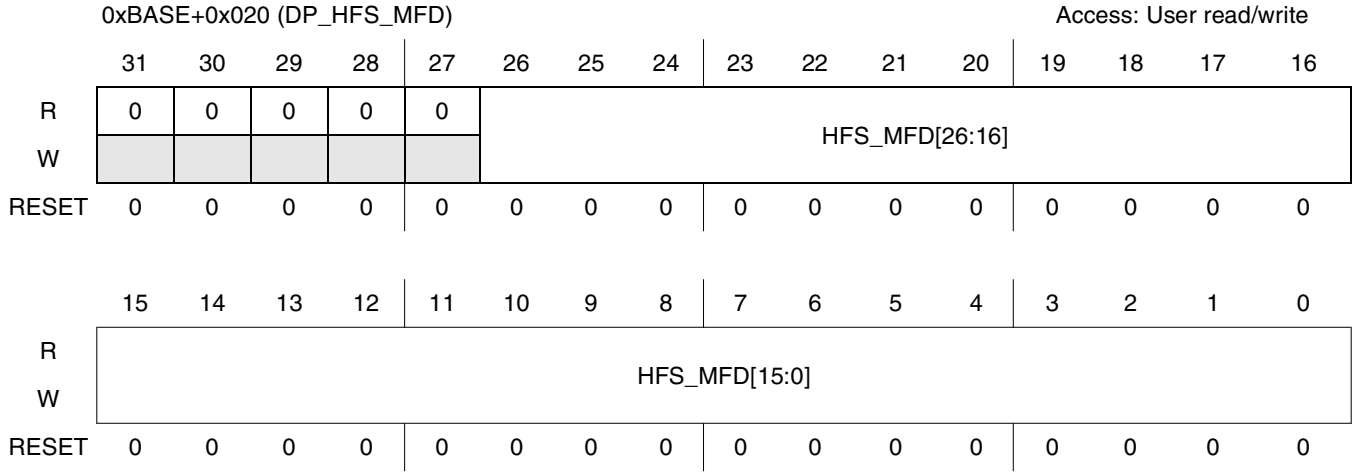


Figure 22-8. DPLL HFS MFD Register Diagram

Table 22-8. DP_HFS_MFD Field Description

Field	Description
31–27	Reserved
26–0 MFD[26:0]	Multiplication Factor Denominator Bits. The bits in the MFD[26:0] field contain the denominator of the fractional part in a 2's compliment format. The range of the MFD is from 0 to 67108863. Settings outside of this range cause the output clock frequency to differ from the desired frequency. **Note:** Bit 26 is always 0. Settings (Hex) 000000 0 000001 1 000002 2 ... 3FFFFE 67108862 3FFFFF 67108863

22.2.2.8 DPLL High Freq. Support MFN Register (DP_HFS_MFN)

Figure 22-6 shows the DP_HFS_MFD register diagram and Table 22-6 provides the field descriptions. The HFS_MFN field is used to program the decimal value of High Frequency Multiplication Factor Numerator which is output to the DPLL module by DPLL-IP.

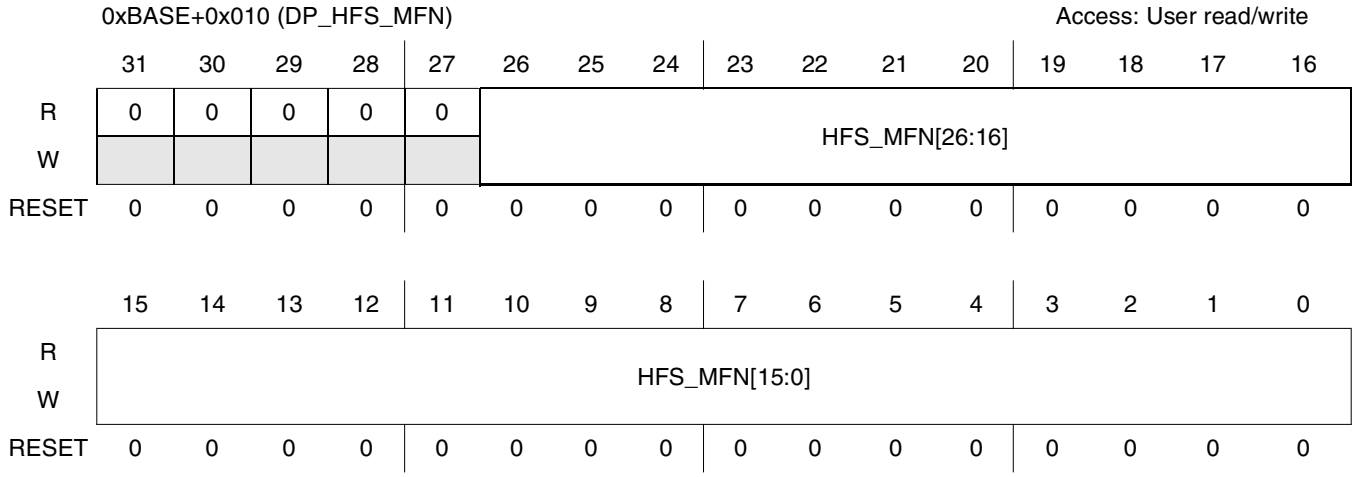


Figure 22-9. DPLL MFNnnn Registers Diagram

Table 22-9. DP_MFNnnn Field Description

Field	Description
26–0 MFNxxx	Multiplication Factor Numerator Bits (nominal, “MINUS”, or “PLUS”)— The MFNxxx[26:0] bits, in a 2’s compliment format, give the numerator of the fractional part. The MFN range is from -67108864d to 67108863d. If the absolute value of the MFN is larger than MFD, the output clock frequency will differ from the desired frequency. The default value of the MFNMINUS and MFNPLUS registers is 0. Settings (Hex) 4000000 -67108864 4000001 -67108863 4000002 -67108862 ... FFFFFFF -1 0000000 0 (default) 0000001 1 ... 3FFFFFE 67108862 3FFFFFF 67108863

22.2.2.9 DPLL High Freq. Support MFN Register (DP_HFS_MFN)

Figure 22-6 shows the DP_HFS_MFD register diagram, and Table 22-6 provides the field descriptions. The HFS_MFN field is used to program the decimal value of High Frequency Multiplication Factor Numerator which is output to the DPLL module by DPLL-IP.

0xBASE+0x010 (DP_HFS_MFN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	HFS_MFN[26:16]										
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HFS_MFN[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-10. DPLL MFNnnn Registers Diagram

22.2.2.10 DPLL Toggle Control Register (DP_MFN_TOGC)

Figure 22-11 shows the DP_MFN_TOGC register diagram and Table 22-10 provides the field descriptions.

0xBASE+0x028 (DP_MFN_TOGC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TOG_DIS	TOG_EN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOG_MFN_CNT[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-11. DPLL Toggle Control Register Diagram

Table 22-10. DP_MFN_TOGC Register Field Descriptions

Field	Description
31–18	**Reserved**
17 TOG_DIS	Desense Disable. If TOG_DIS is set, then the desense logic for the DPLL is disabled, and software (TOG_EN) or hardware (l1t_tog_en) requests to enable the logic are ignored. **Note:** See Table 22-11 for the setting combinations that affect the Desense logic.
16 TOG_EN	Desense On. Software request to activate desense logic. Desense logic may also be activated by a hardware request (l1t_tog_en) **Note:** See Table 22-11 for the setting combinations that affect the Desense logic.
15–0 TOG_CNT[15:0]	toggle counter value. If the value programmed for the count is odd, the even value (original value minus 1) is used. 0000 0 0001 0 0002 2 0003 2 ...

Table 22-11. DP_MFN_TOGC Register Desense Settings

l1t_tog_en	TOG_EN	TOG_DIS	Desense logic
x	x	1	OFF
0	0	0	OFF
0	1	0	on
1	0	0	on
1	1	0	on

22.2.2.11 DPLL Desense Status Register (DP_DESTAT)

Figure 22-12 shows the DP_DESTAT register diagram and Table 22-10 provides the field descriptions.

0xBASE+0x02C (DP_DESTAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TOG_SEL	0	0	0	0	TOG_MFN[26:16]										
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TOG_MFN_CNT[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-12. DPLL Desense Status Register Diagram

Table 22-12. DESENSE STATUS register description

Field	Description
31 TOG_SEL	Toggle sel status. Indicates the status of the desense logic in the DPLL. 0 DPLL desense logic inactive. 1 DPLL desense logic active.
30–27	Reserved
26–0 TOG_MFN [26:0]	This field contains the MFN value that is sent to the DPLL when the desense logic is active.

22.3 Functional Description

22.3.1 Clock Muxing and Gating/div Circuitry

Of the four clocks **clock0**, **clock1**, **clock2** and **clock3** one is selected depending upon the status of **ref_clk_sel**[1:0] signals (bit-8 and bit-9) of the DP_CTL register.

This is followed by a div circuitry and then by a gating circuitry as shown below. If **ref_clk_en_dpllip** signal is HIGH, the clock is passed, else it is gated to a value’1’.

Figure 22-13 illustrates the clock MUXing gating/div circuitry.

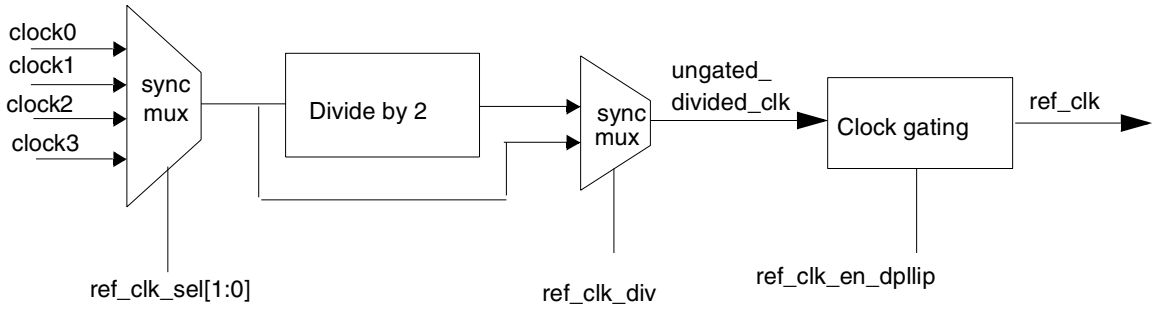


Figure 22-13. Clock MUXing and gating/div Circuitry

22.3.2 DVFS Support: HFS Mode

DPLL supports DVFS settings through an alternate set of registers which can be used to store high frequency settings corresponding to high voltage. This alternate set of HFS registers is used to restart the PLL with high frequency settings stored in HFS registers whenever system detects a voltage change from low to high level. Figure 22-14 describes this configuration.

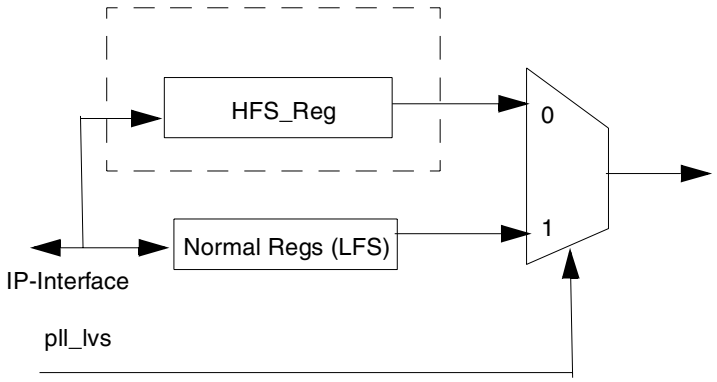


Figure 22-14. DVFS Support for High freq settings

22.3.3 DPLL Control and Port Updating

This section discusses DPLL control and port updating for the DPLL control registers, shadow registers, and DP_MFN register.

22.3.3.1 DP_CTL, DP_OP, DP_MFD Register Update

The DPLLs require special timing considerations when updating their ports. Whenever one of the DPLL control registers DP_CTL, DP_OP, or DP_MFD, or the shadow registers DP_HFS_OP or DP_HFS_MFD is written (even if the bits do not change), a restart is issued to the dpll before the actual control bits are sent to the DPLL. The restart assertion and port update will be synchronous to the reference input clock of the dpll. Writing to any of these registers causes following the procedure to come into effect:

1. Register is written

2. One clock cycle later restart (crstrt) is asserted
3. One clock cycle later the appropriate port is updated
4. Four clock cycles later restart is released

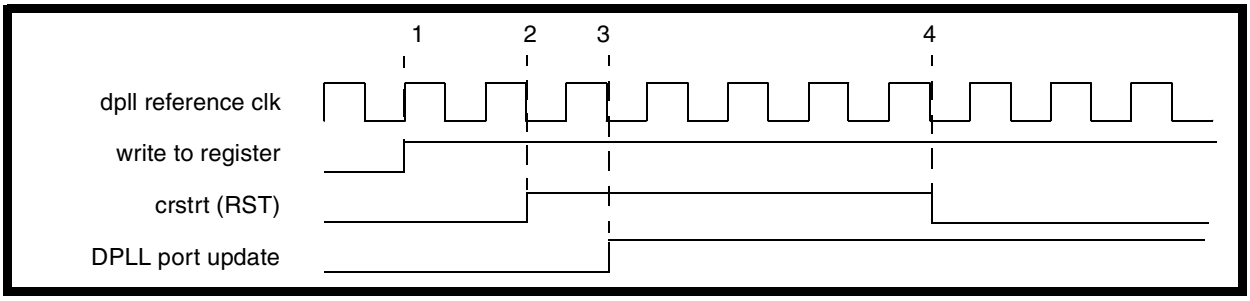


Figure 22-15. DPLL Port Timing

NOTE

Setting the RST bit causes the crstrt signal to immediately assert.

In case of AREN bit is LOW, the programmer will have to take care of restart of dpll by setting the RST bit of DP_CTL register appropriately.

22.3.3.2 DP_MFN Register Update

The MFN bits (in register DP_MFN) are the only bits in the DPLL that can be changed after the DPLL was locked without restart. As shown in Figure 22-16, after writing a new value to MFN, the procedure to update the DPLL is as follows:

1. The control bit LDREQ in DP_CONFIG is set. This sends a request signal to the DPLL.
2. When the DPLL loads the new MFN, its sends an acknowledge signal back to the Interface and clears the write value of LDREQ. At this point LDREQ becomes a status bit, reflecting the state of the acknowledge signal. NOTE: This write value of LDREQ is a “sticky” bit. When set, it will remain at ‘1’ until a load_ack pulse is received.
3. The read value of LDREQ is automatically cleared when the DPLL has completed updating MFN and has removed the acknowledge signal, allowing the next load request to take place.

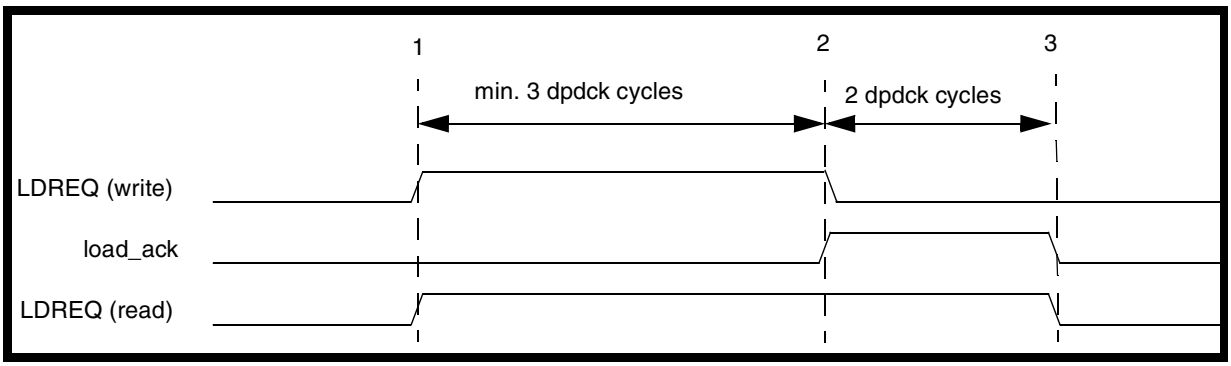


Figure 22-16. MFN Load Request Timing

22.3.4 Multiple Options for DPLL Control

The PLL interface provides multiple options for controlling the ON/OFF logic of the DPLLs from the Core and the DSM, making it possible to switch off the PLL by software and still be able to switch it on by DSM. Mul_ctrl (Bit [13]) of DP_CTL register provides this flexibility as shown in the following table:

Table 22-13. Options for DPLL Control

dpll_en_dpllip	UPEN bit	mul_ctrl bit (Self Clearing)	dpllip_cpen
X	X	1	0
X	0	X	0
posedge	1	0	1
0	X	X	0

X = Don't Care

As shown in Table 22-3, once the DPLL is switched off by Software using the Mul_ctrl bit, the software cannot turn the PLL on thereafter,unless there is a posedge on the dpll_en_dpllip signal, for example during the DSM entry sequence.

The dpllip_cpen signal shown above is the internally generated cpen signal which is then muxed with external signals to generate the final enable signal to the DPLL,

22.4 Initialization/Application Information

DPLL programming codes are shown below. In the examples, DPLL has been programmed to output frequency at 156 MHz (Example 22-1), 208 MHz (Example 22-2), and 400 MHz (Example 22-3).

Example 22-1. Example Code for DPLL Output = 156 MHz, with Reference Clock = 26 MHz:

```
// ***** Transfer ownership by writing to RAR[2:0] bits in SPBA PRR register
        // Write RAR as 111 giving access permission to all 3 masters(AP, BP, SDMA)
        mem32_write(SPBA_PLL1_PRR, 0x00000007);

         // It can then read back the RMO, RAR, ROI bits to make sure RMO returns the right
value, ROI bits contain its ID and RAR bits are correctly asserted.
         // The value should be 0xc0010007 (for ARM Access) and 0xc0020007(for DSP Access)
        if (mem32_read(SPBA_PLL1_PRR) == 0xc0020007)
                    {verilog_trigger(vt_pass);}
            else {verilog_trigger(vt_fail);}


//***** Program the PLL1 to lock at 156 MHz

      // Program DPLL -IPs for auto restart enable: setting AREN bit in PLL_DP_CONFIG register //
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

       // Assert soft reset to DPLLs through DPLL-IP : setting BRMO and PRE bits in PLL_DP_CTL
registers //
        mem32_write( PLL1_DP_CTL , 0x00000042);
        mem32_write( PLL1_DP_CTL , 0x00000000);
```

```
// **** Program registers to generate freq 156MHz from PLL//

      // Program DPLL -IPs for auto restart enable : setting AREN bit  in PLL_DP_CONFIG register
//
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

        // Writing 1 to PDF(divide by 2) and 6 to MFI(MFI = 6)
        mem32_write( PLL1_DP_HFS_OP , 0x00000061);

       // Writing MFN as 0
        mem32_write( PLL1_DP_HFS_MFN , 0x00000000);

       // Writing MFD as 0
        mem32_write( PLL1_DP_HFS_MFD , 0x00000000);

        //  Writing to PLL_DP_CTL register : Setting BRMO and UPEN bits
        mem32_write( PLL1_DP_CTL , 0x00000022);

        // Wait for  PLL to be locked : checking the lrf flag//
        while((mem32_read(PLL1_DP_CTL) & 0x00000001) != 0x00000001);

The same can be found in /vobs/vb_scm_a11/scm_a11_settings/testbench/p2002/src/main.c
```

Example 22-2. Example Code for DPLL Output = 208 MHz, with Reference Clock = 16.8 MHz

```
// ***** Transfer ownership by writing to RAR[2:0] bits in SPBA PRR register
        // Write RAR as 111 giving access permission to all 3 masters(AP, BP, SDMA)
        mem32_write( SPBA_PLL1_PRR , 0x00000007);

         // It can then read back the RMO, RAR, ROI bits to make sure RMO returns the right
value, ROI bits contain its ID and RAR bits are correctly asserted.
         // The value should be 0xc0010007 (for ARM Access) and 0xc0020007(for DSP Access)
if (mem32_read(SPBA_PLL1_PRR) == 0xc0020007)
                    { verilog_trigger(vt_pass); }
            else { verilog_trigger(vt_fail); }


//***** Program the PLL1 to lock at 208 MHz

      // Program DPLL -IPs for auto restart enable : setting AREN bit  in PLL_DP_CONFIG register
//
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

       // Assert soft reset to DPLLs through DPLL-IP : setting BRMO and PRE bits in PLL_DP_CTL
registers //
        mem32_write( PLL1_DP_CTL , 0x00000042);
        mem32_write( PLL1_DP_CTL , 0x00000000);

        // **** Program registers to generate freq 208MHz from PLL//

      // Program DPLL -IPs for auto restart enable : setting AREN bit  in PLL_DP_CONFIG register
//
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

        // Writing 0 to PDF(divide by 1) and 6 to MFI(MFI = 6)
```

```
        mem32_write( PLL1_DP_HFS_OP , 0x00000060);

       // Writing MFN as 190476
        mem32_write( PLL1_DP_HFS_MFN , 0x0002e80c);

       // Writing MFD as 999999
        mem32_write( PLL1_DP_HFS_MFD ,  0x000f423f);
        //  Writing to PLL_DP_CTL register : Setting BRMO and UPEN bits


mem32_write( PLL1_DP_CTL , 0x00000022);

        // Wait for  PLL to be locked : checking the lrf flag//
        while((mem32_read(PLL1_DP_CTL) & 0x00000001) != 0x00000001);
```

Example 22-3. Example Code for DPLL Output = 400 MHz, with Reference Clock = 16.8 MHz

```
// ***** Transfer ownership by writing to RAR[2:0] bits in SPBA PRR register
        // Write RAR as 111 giving access permission to all 3 masters(AP, BP, SDMA)
        mem32_write( SPBA_PLL1_PRR , 0x00000007);

         // It can then read back the RMO, RAR, ROI bits to make sure RMO returns the right
value, ROI bits contain its ID and RAR bits are correctly asserted.
         // The value should be 0xc0010007 (for ARM Access) and 0xc0020007(for DSP Access)
        if (mem32_read(SPBA_PLL1_PRR) == 0xc0020007)
                    { verilog_trigger(vt_pass); }
            else { verilog_trigger(vt_fail); }


//***** Program the PLL1 to lock at 400 MHz

     // Program DPLL -IPs for auto restart enable : setting AREN bit  in PLL_DP_CONFIG register
//
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

       // Assert soft reset to DPLLs through DPLL-IP : setting BRMO and PRE bits in PLL_DP_CTL
registers //
        mem32_write( PLL1_DP_CTL , 0x00000042);
        mem32_write( PLL1_DP_CTL , 0x00000000);


// **** Program registers to generate freq 400MHz from PLL//

// Program DPLL -IPs for auto restart enable : setting AREN bit  in PLL_DP_CONFIG register //
        mem32_write( PLL1_DP_CONFIG , 0x00000002);

        // Writing 0 to PDF(divide by 1) and 11 to MFI(MFI = 11)
        mem32_write( PLL1_DP_HFS_OP , 0x000000b0);

       // Writing MFN as 904762
        mem32_write( PLL1_DP_HFS_MFN , 0x000dce3a);

       // Writing MFD as 999999
        mem32_write( PLL1_DP_HFS_MFD ,  0x000f423f);

        //  Writing to PLL_DP_CTL register : Setting BRMO and UPEN bits
```

```
mem32_write( PLL1_DP_CTL , 0x00000022);

// Wait for  PLL to be locked : checking the lrf flag//
while((mem32_read(PLL1_DP_CTL) & 0x00000001) != 0x00000001);
```

Chapter 23
Dynamic Process and Temperature Compensation (DPTC)

23.1 Introduction

The DPTC (Dynamic Process and Temperature Compensation) module is a power management module. The purpose of the DPTC module is to detect the minimum operation voltage for the IC, regarding the process corner case and temperature for a given frequency. It obtains predefined values for process speed performance measurement and generates interrupt, if supply voltage value update is required.

NOTE

DPTC is a monitor that only provides an interrupt when counting exceeds a predefined value and does not actually send requests to change the voltage. This can be done by the user in the interrupt routine or SDMA code by using GPC or CCM.

23.1.1 Features

The DPTC features include the following:

- Interrupt generation on performance limit violation. Such an interrupt causes voltage value update.
- DPTC disabling at interrupt routine execution (by DPNVCR bit)
- DPTC auto re-enabling when requested voltage achieved (by PMIC_RDY signal posedge)
- DPTC disabling/enabling when DSM mode is active/ended.
- Four reference circuits support

23.1.2 Debug Mode

The debug mode is activated by writing ‘1’ to the RCLKON bit. The first activated reference circuit continues to run endlessly. DPTC interrupt should be masked.

23.2 Memory Map and Register Definition

23.2.1 DPTC Memory Map

The addresses are staring [11:2] = 01_0000_0000 value.

Table 23-1. AP Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE_$0000 (DPTCCR)	DPTC Control Register (DPTCCR)	R/W	0x0004_0038	23.2.4.1/23-4
0xBASE_$0004 (DPTCDBG)	DPTC Debug Register (DPTCDBG)	R/W	0x0000_0000	23.2.4.2/23-6
0xBASE_$0008 (DCVR0)	DPTC Comparator Value Register0 (DCVR0)	R/W	0x0000_0000	23.2.4.3/23-7
0xBASE_$000C (DCVR1)	DPTC Comparator Value Register1 (DCVR1)	R/W	0x0000_0000	23.2.4.3/23-7
0xBASE_$0010 (DCVR2)	DPTC Comparator Value Register2 (DCVR2)	R/W	0x0000_0000	23.2.4.3/23-7
0xBASE_$0014 (DCVR3)	DPTC Comparator Value Register3 (DCVR3)	R/W	0x0000_0000	23.2.4.3/23-7

23.2.2 Register Summary

The conventions in Figure 1-1 and Table 1-7 serve as a key for the register summary and individual register diagrams.

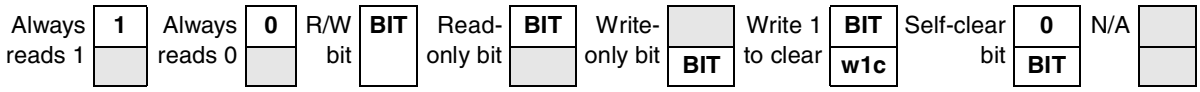


Figure 23-1. Key to Register Fields

Table 1-7 provides a key for register figures and tables and the register summary.

Table 23-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.

Table 23-2. Register Conventions (continued)

Convention	Description
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

23.2.3 DPTC Register Summary

Table 23-3 shows the DPTC register summary.

Table 23-3. DPTC Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE_$000 (DPTCCR)	R	0	0	0	0	0	0	0	0	0	DRCE3	DRCE2	DRCE1	DRCE0	0	DCR	0
	W																
	R	0	0	0	0	0	0	0	0	0	DSMM	DP-NVCR	DPVV	VAIM	VAI[1:0]		DEN
	W																
0xBASE_$004 (DPTCDBG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	RCLKON	0	RCCR											RC-CRV
	W																
0xBASE_$008 (DCVR0)	R	ULV											LLV				
	W																
	R	LLV						ELV									
	W																
0xBASE_$00C (DCVR1)	R	ULV											LLV				
	W																
	R	LLV						ELV									
	W																
0xBASE_$010 (DCVR2)	R	ULV											LLV				
	W																
	R	LLV						ELV									
	W																

Table 23-3. DPTC Register Summary (continued)

Name		31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0xBASE_$014 (DCVR3)	R	ULV LLV
	W	
	R	LLV ELV
	W	

23.2.4 Register Descriptions

This section shows the individual register descriptions.

23.2.4.1 DPTC Control Register (DPTCCR)

Figure 23-2 shows the DPTC control register, and Table 23-4 describes its fields.

0xBASE_$000 (DTPCCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	DRCE3	DRCE2	DRCE1	DRCE0	DCR		0
W																
RESET:										0	0	0	0	10		

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	DSMM	DPN-VCR	DPVV	VAIM	VAI1	VAI0	DEN
W																
RESET:										0	1	1	1			0

Figure 23-2. DPTC Control Register (DPTCCR)

Table 23-4. Register Field Descriptions

Field	Description
DRCE3	DPTC reference circuit 3 enable bits. This bit defines whether reference circuit 3 is enabled during DPTC operation. 1 DPTC reference circuit 3 enabled. 0 DPTC reference circuit 3 disabled. **Note:** Write to this bit is permitted only when DPTC is disabled.
DRCE2	DPTC reference circuit 2 enable bits. This bit defines whether reference circuit 2 is enabled during DPTC operation. 1 DPTC reference circuit 2 enabled. 0 DPTC reference circuit 2 disabled. **Note:** Write to this bit is permitted only when DPTC is disabled.

Table 23-4. Register Field Descriptions (continued)

Field	Description
DRCE1	DPTC reference circuit 1 enable bits. This bit defines whether reference circuit 1 is enabled during DPTC operation. 1 DPTC reference circuit 1 enabled. 0 DPTC reference circuit 1 disabled. **Note:** Write to this bit is permitted only when DPTC is disabled.
DRCE0	DPTC reference circuit 0 enable bits. This bit defines whether reference circuit 0 is enabled during DPTC operation. 1 DPTC reference circuit 0 enabled. 0 DPTC reference circuit 0 disabled. **Note:** Write to this bit is permitted only when DPTC is disabled.
DCR	DPTC counting range. This bit configures a number of system clocks, during which the reference circuits will be active (and their output signals will be counted). Value of '1' causes 256 system clock count. Value of '0' causes 128 system clock count. 11 256 system clock count 10 128 system clock count 01 64 system clock count 00 32 system clock count **Note:** Write to this bit is permitted only when DPTC is disabled.
DSMM	IPG_STOP input masking. This masking prevents DPTC disabling when 'ipg_stop' signal goes high. 'ipg_stop' signal goes low at system entering DSM mode (sys_clk of DPTC will disappear). 1 IPG_STOP is masked 0 IPG_STOP is not masked
DPNVCR	DPTC No Voltage Change request. Bit is written by SW only. 1 DPTC enabled 0 DPTC disabled
DPVV	DPTC voltage valid edge detect. Can be overwritten by SW. Bit is cleared when DPNVCR is cleared (upon its negedge) or when 'pmic_rdy' input goes low (upon its negedge). Bit is set when 'pmic_rdy' goes high (upon its posedge). 1 Received a voltage valid acknowledge 0 Voltage is not valid
VAIM	Voltage adjustment interrupt mask. This bit masks voltage adjustment interrupt. VAI status bits will be still asserted in relevant case. 1 Interrupt is masked. 0 Interrupt is enabled.
VAI [1:0]	Voltage adjustment status. This status bits indicate, that supply voltage should change. Bits are cleared when DPTC is disabled or if limits are not violated. Read only. 00 No interrupt 01 High-limit indication. Voltage should be decreased. Interrupt is asserted, if VAIM=0 10 Low-limit indication. Voltage should be increased. Interrupt is asserted, if VAIM=0 11 Emergency-limit indication. Voltage should be increased immediately. Interrupt is asserted if VAIM=0.
DEN	DPTC enable. This bit enables DPTC block, starts reference circuit counting and compares results to the look-up table values. DPTC is enabled only if DEN = 1, DPVV = 1, DPNVCR = 1, IPG_STOP = 0 (or masked). 1 DPTC is enabled. 0 DPTC is disabled.

23.2.4.2 DPTC Debug Register (DPTCDBG)

Figure 23-3 shows the DPTC debug register, and Table 23-5 describes its fields.

0xBASE_$004 (DPTCDBG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET:																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	RCLKON	0	RCCR											RC-CRV
W																
RESET:			0													

Figure 23-3. DPTC Debug Register (DPTCDBG)

Table 23-5. Register Field Descriptions

Field	Description
RCLKON	Reference circuit clock on. Used for DPTC debugging for viewing on CKO2 pin. If bit is set, DPTC interrupt has to be masked. 1 DPTC Debug mode—first selected reference circuit clock is always running, and other reference circuits are not selected. 0 Normal operation of DPTC. **Note:** Write to this bit is permitted only when DPTC is disabled.
RCCR	Reference Circuit Counter Result. Holds first result that was read from selected reference circuit. Bits are cleared when DPTC is disabled (DEN=0). Read-only bits.
RCCRV	Reference Circuit Counter Result Valid. Indicates that result in RCCR bits is valid. Bit is cleared when DPTC is disabled (DEN=0). Read-only status bit. 1 Result in RCCR is valid 0 Result in RCCR is not valid.

23.2.4.3 DPTC Comparator Value Registers 0–3 (DCVR0–3)

Figure 23-4 shows the DPTC comparator value registers 0–3, and Table 23-6 describes their fields.

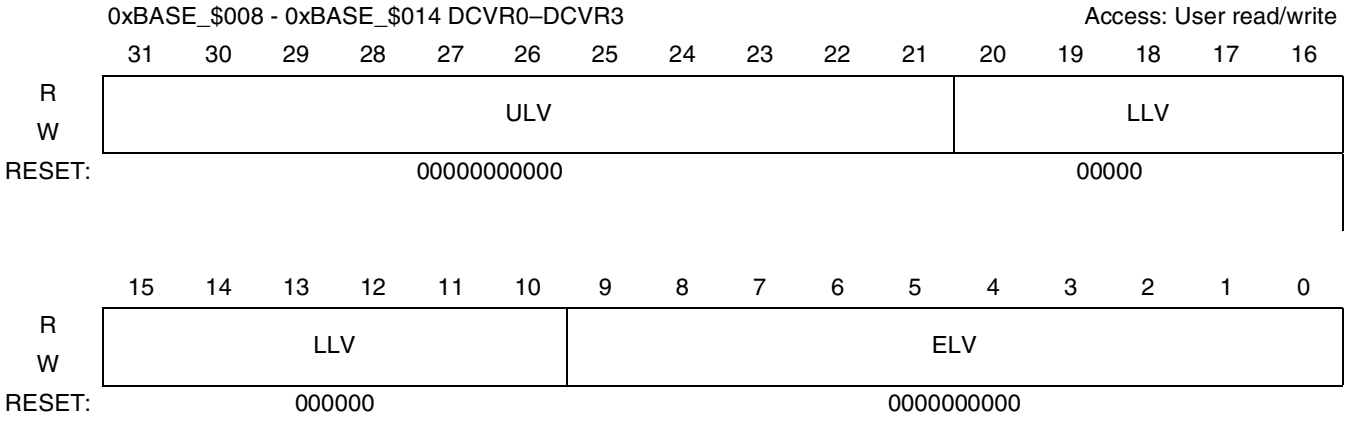


Figure 23-4. DPTC Comparator Value Registers 0–3 (DCVR0–3)

Table 23-6. Register Field Descriptions

Field	Description
ULV	Upper Limit Value (11 bits)—value of reference circuit clock counts for upper performance limit
LLV	Lower Limit Value (11 bits)—value of reference circuit clock counts for lower performance limit.
ELV	Emergency Limit Value (10 bits)—value of reference circuit clock counts for lower emergency performance limit. Emergency Limit Value should be less than Lower Limit Value. **Note:** For ELV field, Bit 9 is used as MSBit in DPTC (b10). Bit 0 is extended for LSBit in DPTC. Full emergency limit value (11 bit) should be less than lower limit value.

These registers contain relevant DPTC Look-up table values. Each register hold the limit values for the corresponding reference circuit (for example, DCVR0 corresponds to reference circuit 0). Upper limit should be always higher than lower limit. Lower limit should be always higher than emergency limit.

Chapter 24
Dynamic Voltage Frequency Scaling (DVFS)

24.1 Introduction

The DVFS allows simple dynamic voltage frequency scaling. The frequency of the core clock domain and the voltage of the core power domain can be changed on the fly while all modules (including the MCU) continue their normal operation. The frequency of the core clock domain can be changed by switching temporally to an alternate PLL clock, and then return to the updated PLL, already locked at a specific frequency, or by merely changing the post dividers division factors.

The DVFS load tracking block allows hardware tracking on the core load and a generation of an interrupt when a frequency change is requested.

NOTE

DVFS is a monitor that only provides an interrupt when counting exceeds the predefined value and does not actually send the request to make a change of voltage and frequency. This can be done by the user in the interrupt routine or SDMA code by using GPC or CCM (relocking the PLL or changing the post dividers at the CCM).

24.1.1 Overview

Figure 24-1 shows the block diagram.

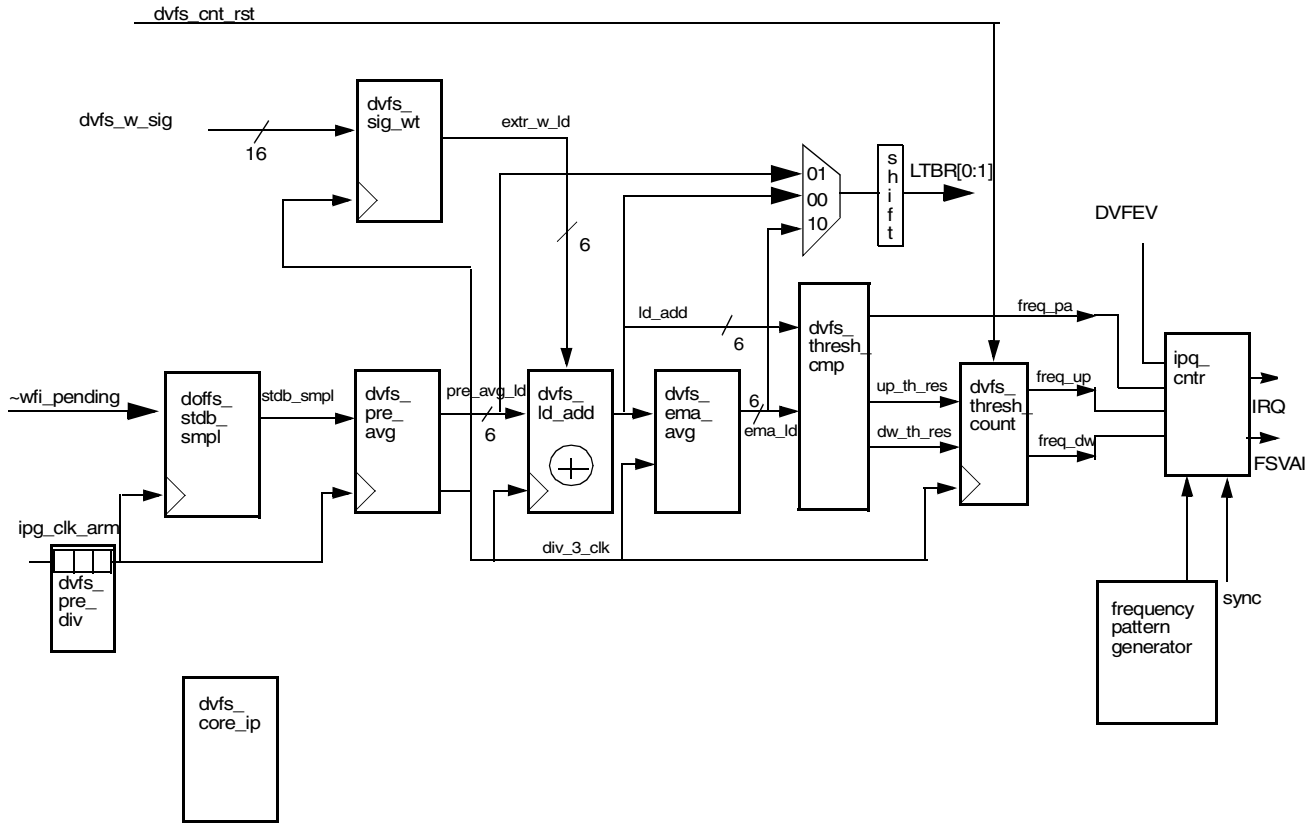


Figure 24-1. Block Diagram

24.1.2 Features

The DVFS load tracking block includes the following features:

- Configurable include/exclude of input signals:
  - — ARM standby signal (idle / non-idle)
  - — 16 general-purpose bits
    - – Configurable weight of each bit.
- Configurable generated clocks and averaging time slicing (respond time).
- Configurable panic mode respond logic (for frequency up).
- Programmable buffer for last 4,8,12, or 16 load tracking samples. Based on value in LBCF in DVFSCNTR. There is also a buffer full signal, LBFL.

24.2 Memory Map and Register Definition

This section provides a memory map, register summary, and register definitions.

24.2.1 Memory Map

Table 24-3 shows the memory map.

Table 24-3. Block Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE_0000	DVFSTHRS—DVFS Thresholds	R/W[1]	0x0FAF_003E	24.2.3.1/24-7
0xBASE_0004	DVFSCOUN—DVFS Counters thresholds	R/W	0x0007_0020	24.2.3.2/24-7
0xBASE_0008	DVFSSIG1—DVFS general purpose bits weight	R/W	0x0000_0000	24.2.3.3/24-8
0xBASE_000c	DVFSSIG0—DVFS general purpose bits weight	R/W	0x0000_0000	24.2.3.4/24-9
0xBASE_0010	DVFSGPC0—DVFS general purpose bit 0 weight counter	R/W	0x0000_0000	24.2.3.5/24-10
0xBASE_0014	DVFSGPC1—DVFS general purpose bit 1 weight counter	R/W	0x0000_0000	24.2.3.6/24-10
0xBASE_0018	DVFSGPBT—DVFS general purpose bits enable	R/W	0x0000_0000	24.2.3.7/24-11
0xBASE_001C	DVFSEMAC—DVFS EMAC settings	R/W	0x0000_0004	24.2.3.8/24-12
0xBASE_0020	DVFSCNTR—DVFS Control	R/W	0x0900_000E	24.2.3.9/24-13
0xBASE_0024	DVFSLTR0_0—DVFS Load Tracking Register 0, portion 0	R	0x0000_0000	24.2.3.10/24-16
0xBASE_0028	DVFSLTR0_1—DVFS Load Tracking Register 0, portion 1	R	0x0000_0000	24.2.3.11/24-16
0xBASE_002C	DVFSLTR1_0—DVFS Load Tracking Register 1, portion 0	R	0x0000_0000	24.2.3.12/24-17
0xBASE_0030	DVFSLTR1_1—DVFS Load Tracking Register 3, portion 1	R	0x0000_0000	24.2.3.13/24-18
0xBASE_0034	DVFSPT0—DVFS pattern 0 length	R/W	0x0000_0010	24.2.3.14/24-18
0xBASE_0038	DVFSPT1—DVFS pattern 1 length	R/W	0x0000_0010	24.2.3.15/24-19
0xBASE_003c	DVFSPT2—DVFS pattern 2 length	R/W	0x0000_0010	24.2.3.16/24-19
0xBASE_0040	DVFSPT3—DVFS pattern 3 length	R/W	0x0000_0010	24.2.3.17/24-20

[1] Note that R/W registers may contain some read-only or write-only bits.

24.2.2 Register Summary

The conventions in Figure 24-2 and Table 24-4 serve as a key for the register summary and individual register diagrams.

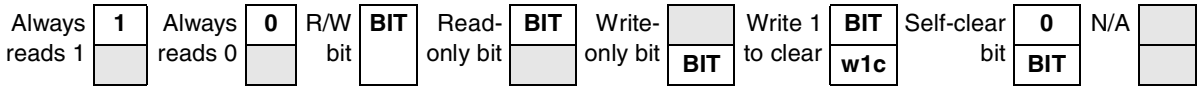


Figure 24-2. Key to Register Fields

Table 24-4. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 24-5 shows the DVFS register summary table.

Table 24-5. Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR+ 0000 DVFSTHRS	R	0	0	0	0	UPTHR						DNTHR					
	W																
	R	0	C	0	0	0	0	0	0	0	0	PNCTHR					
	W																
BASE_ADDR + 0004 DVFSCOUN	R	0	0	0	0	0	0	0	0	DWCNT							
	W																
	R	0	C	0	0	0	0	0	0	UPCNT							
	W																
BASE_ADDR + 0008 DVFSSIG1	R	WSW15			WSW14			WSW13			WSW12			WSW11			WS W10
	W																
	R	WSW10			WSW9			WSW8			WSW7			WSW6			
	W																

Table 24-5. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR + 000c DVFSSIG0	R	WSW5			WSW4			WSW3			WSW2						
	W																
	R					WSW1						WSW0					
	W																
BASE_ADDR + 0010 DVFSGPC0	R	C0STRT	C0ACT														GPBC0[16]
	W																
	R	GPBC0[15:0]															
	W																
BASE_ADDR + 0014 DVFSGPC1	R	C1STRT	C1ACT														GPBC1[16]
	W																
	R	GPBC1[15:0]															
	W																
BASE_ADDR + 0018 DVFSGPBT	R																
	W																
	R	GPB15	GPB14	GPB13	GPB12	GPB11	GPB10	GPB9	GPB8	GPB7	GPB6	GPB5	GPB4	GPB3	GPB2	GPB1	GPB0
	W																
BASE_ADDR + 001C DVFSEMAC	R																
	W																
	R								EMAC								
	W																
BASE_ADDR + 0020 DVFSCNTR	R	DIV3CK			DVFEV	LBMI	LBFL1	LBFL0	DVFIS	PIRQS	FSVAIM	FSVAI		WFIM	MAXF	MINF	
	W																
	R	DIV_RATIO						PFUE	PFUS			LT-BRSH	LT-BRSR				DVFEN
	W																
BASE_ADDR + 0024 DVFSLTR0_0	R	LTS0_7				LTS0_6				LTS0_5				LTS0_4			
	W																
	R	LTS0_3				LTS0_2				LTS0_1				LTS0_0			
	W																

Table 24-5. Register Summary (continued)

Name		31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16															
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR + 0028 DVFSLTR0_1	R	LTS0_15				LTS0_14				LTS0_13				LTS0_12			
	W																
	R	LTS0_11				LTS0_10				LTS0_9				LTS0_8			
	W																
BASE_ADDR + 002C DVFSLTR1_0	R	LTS1_7				LTS1_6				LTS1_5				LTS1_4			
	W																
	R	LTS1_3				LTS1_2				LTS1_1				LTS1_0			
	W																
BASE_ADDR + 0030 DVFSLTR1_1	R	LTS1_15				LTS1_14				LTS1_13				LTS1_12			
	W																
	R	LTS1_11				LTS1_10				LTS1_9				LTS1_8			
	W																
BASE_ADDR + 0034 DVFSPT0	R															PT0A	FPTN0[16]
	W																
	R	FPTN0[15:0]															
	W																
BASE_ADDR + 0038 DVFSPT1	R															PT1A	FPTN1[16]
	W																
	R	FPTN1[15:0]															
	W																
BASE_ADDR + 003C DVFSPT2	R															PT2A	FPTN2[16]
	W																
	R	FPTN2[15:0]															
	W																
BASE_ADDR + 0040 DVFSPT3	R															PT3A	FPTN3[16]
	W																
	R	FPTN3[15:0]															
	W																

24.2.3 Register Descriptions

This section provides detailed descriptions of the DVFS registers. The descriptions are in address order.

24.2.3.1 DVFSTHRS Register

Figure 24-3 shows the DVFSTHRS register, and Table 24-6 describes the register fields.

Address 0xBASE_0000 (DVFSTHRS) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	UPTHR						DNTHR					
W																
Reset	0	0	0	0	1	1	1	1	1	0	1	0	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	C	0	0	0	0	0	0	0	0	PNCTHR					
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0

3

Figure 24-3. DVFSTHRS Register

Table 24-6. DVFSTHRS Register Field Descriptions

Field	Description
27–22 UPTHR	Upper threshold for load tracking
21–16 DWTHR	Down threshold for load tracking
5–0 PNCTHR	Panic threshold for load tracking

24.2.3.2 DVFSCOUN Register

Figure 24-4 shows the DVFSCOUN register, and Table 24-7 describes its register fields.

Address 0xBASE_0004 (DVFSCOUN) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	DNCNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	C	0	0	0	0	0	0	UPCNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

4

Figure 24-4. DVFSCOUN

Table 24-7. DVFSCOUN Register Field Descriptions

Field	Description
23–16 DN CNT	Down counter threshold value
7–0 UPCNT	UP counter threshold value

24.2.3.3 DVFSSIG1 Register

Figure 24-4 shows the DVFSSIG1 register, and Table 24-8 describes its register fields.

Address 0xBASE_0008 (DVFSSIG1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WSW15				WSW14				WSW13				WSW12			WSW 10
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WSW10			WSW9			WSW8		WSW7			WSW6				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-5. DVFSSIG1 Register

Table 24-8. DVFSSIG1 Register Field Descriptions

Field	Description
31–29 WSW15	General purpose load tracking signal weight dvfs_w_sig[15]
28–26 WSW14	General purpose load tracking signal weight dvfs_w_sig[14]
25–23 WSW13	General purpose load tracking signal weight dvfs_w_sig[13]
22–20 WSW12	General purpose load tracking signal weight dvfs_w_sig[12]
19–17 WSW11	General purpose load tracking signal weight dvfs_w_sig[11]
16–14 WSW10	General purpose load tracking signal weight dvfs_w_sig[10]
13–11 WSW9	General purpose load tracking signal weight dvfs_w_sig[9]
10–8 WSW8	General purpose load tracking signal weight dvfs_w_sig[8]

Table 24-8. DVFSSIG1 Register Field Descriptions (continued)

Field	Description
7–5 WSW7	General purpose load tracking signal weight dvfs_w_sig[7]
4–2 WSW6	General purpose load tracking signal weight dvfs_w_sig[6]

24.2.3.4 DVFSSIG0 Register

Figure 24-6 shows the DVFSSIG0 register, and Table 24-9 describes the register fields.

Address 0xBASE_000C (DVFSSIG0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WSW5				WSW4				WSW3				WSW2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					WSW1						WSW0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

5

Figure 24-6. DVFSSIG0 Register

Table 24-9. DVFSSIG0 Register Field Descriptions

Field	Description
31–29 WSW5	General purpose load tracking signal weight dvfs_w_sig[5]
28–26 WSW4	General purpose load tracking signal weight dvfs_w_sig[4]
25–23 WSW3	General purpose load tracking signal weight dvfs_w_sig[3]
22–20 WSW2	General purpose load tracking signal weight dvfs_w_sig[2]
19–12	Reserved
11–6 WSW1	General purpose load tracking signal weight dvfs_w_sig[1]. This value is relevant during GPC1 counting period or when GPB1 is set.
5–0 WSW0	General purpose load tracking signal weight dvfs_w_sig[0]. This value is relevant during GPC0 counting period or when GPB0 is set.

24.2.3.5 DVFSGPC0 Register

Figure 24-7 shows the DVFSGPC0 register, and Table 24-10 describes the register fields.

Address 0xBASE_0010 (DVFSGPC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	C0STRT	C0ACT															GPBC0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-7. DVFS General Purpose Bits Weight Counter

Table 24-10. DVFSGPC0 Register Field Descriptions

Field	Description
31 C0STRT	C0STRT—Counter 0 start Setting of this bit will initialize down counting of the GPC0 value. Bit is self-cleared next cycle after setting. Any setting of this bit will re-start GPC0 counter to the GPC0 value. GPB0 bit disables (overrides) GPC0 counter - WSW0 weight is applicable continuously
30 C0ACT	C0ACT—Counter 0 active indicator 1 General Purpose bit0 counter didn't reach value of "0"—the WSW0 is provided to DVFS calculation 0 General Purpose bit0 counter reached value of "0" - the instead of WSW0, "0" (zero) is provided to DVFS calculation
29–17	Reserved
16–0 GPBC0	GPBC0—General Purpose bits counter 0 During the period of this counter, the GeP bit 0 is set and WSW0 is added to the calculations.

24.2.3.6 DVFSGPC1 Register

Figure 24-8 shows the DVFSGPC1 register, and Table 24-11 describes its register fields.

Address 0xBASE_0014 (DVFSGPC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	C1STRT	C1ACT															GPBC1															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-8. DVFS General Purpose Bits Weight Counter 1

Table 24-11. DVFSGPC1 Register Field Descriptions

Field	Description
31 C1STRT	C1STRT—Counter 1 start Setting of this bit will initialize down counting of the GPC1 value. Bit is self-cleared next cycle after setting. Any setting of this bit will re-start GPC1 counter to the GPC1 value. GPB1 bit disables (overrides) GPC1 counter - WSW1 weight is applicable continuously
30 C1ACT	C1ACT—Counter 1 active indicator 1 - General Purpose bit1 counter didn't reach value of "0" - the WSW1 is provided to DVFS calculation 0 - General Purpose bit1 counter reached value of "0" -instead of WSW1, "0" (zero) is provided to DVFS calculation
29–17	Reserved
16–0 GPBC1	GPBC1—General Purpose Bits Counter 1 During period of this counter the GeP bit 1 will be set and WSW1 will be added to the calculations.

24.2.3.7 DVFSGPBT Register

Figure 24-9 shows the DVFSGPBT register, and Table 24-12 describes its register fields.

Address 0xBASE_0018 (DVFSGPBT) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPB15	GPB14	GPB13	GPB12	GPB11	GPB10	GPB9	GPB8	GPB7	GPB6	GPB5	GPB4	GPB3	GPB2	GPB1	GPB0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-9. DVFSGPBT Register

Table 24-12. DVFSGPBT Register Field Descriptions

Field	Description
31–16 reserved	N/A
15 GPB15	General purpose bit 15. Its weight is set by WSW15 value.
14 GPB14	General purpose bit 14. Its weight is set by WSW14 value.
13 GPB13	General purpose bit 13. Its weight is set by WSW13 value.
12 GPB12	General purpose bit 12. Its weight is set by WSW12 value.
11 GPB11	General purpose bit 11. Its weight is set by WSW11 value.

Table 24-12. DVFSGPBT Register Field Descriptions (continued)

Field	Description
10 GPB10	General purpose bit 10. Its weight is set by WSW10 value.
9 GPB9	General purpose bit 9. Its weight is set by WSW9 value.
8 GPB8	General purpose bit 8. Its weight is set by WSW8 value.
7 GPB7	General purpose bit 7. Its weight is set by WSW7 value.
6 GPB6	General purpose bit 6. Its weight is set by WSW6 value.
5 GPB5	General purpose bit 5. Its weight is set by WSW5 value.
4 GPB4	General purpose bit 4. Its weight is set by WSW4 value.
3 GPB3	General purpose bit 3. Its weight is set by WSW3 value.
2 GPB2	General purpose bit 2. Its weight is set by WSW2 value.
1 GPB1	General purpose bit 1. Its weight is set by WSW1 value. IF set (1), the GPBC1 operation is disregarded, WSW1 value is applied continuously.
0 GPB0	General purpose bit 0. Its weight is set by WSW0 value. IF set (1), the GPBC0 operation is disregarded, WSW0 value is applied continuously.

24.2.3.8 DVFSEMAC Register

Figure 24-10 shows the DVFSEMAC register, and Table 24-13 describes its register fields.

Address 0xBASE_001C (DVFSEMAC) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								EMAC								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

6

Figure 24-10. DVFSEMAC Register

Table 24-13. DVFSEMAC Register Field Descriptions

Field	Description
31–9 reserved	Reserved
8–0 EMAC	EMAC—EMA control value

24.2.3.9 DVFSCNTR Register

Figure 24-11 shows the DVFSCNTR register, and Table 24-14 describes its register fields.

Address 0xBASE_0020 (DVFSCNTR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DIV3CK			DVFEV	LBMI	LBFL1	LBFL0	DVFIS	PIRQS	FSVAIM	FSVAI		WFIM	MAXF	MINF	
Reset	1	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DIV_RATIO						PFUE	PFUS			LT-BRSH	LT-BRSR				DVFEN
Reset	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

Figure 24-11. DVFSCNTR Register

Table 24-14. DVFSCNTR Register Field Descriptions

Field	Description
31–29 DIV3CK	DIV3CK - div_3_clk division ratio inside the DVFS module. According to the Table 24-15
28 DVFEV	Always give a DVFS event. 0 - Do not give an event always. 1- Always give event.
27 LBMI	Load buffer full mask interrupt. This bit masks the generation of this interrupt. 1 = Load buffer full interrupt is masked (LBFL0 and LBFL1 bits still are updated, but interrupt won't be generated) 0 = Load buffer full interrupt is enabled.
26 LBFL1	Load buffer 1 - full status bit. This bit indicates that log buffer registers are full. The bit is set to 1 automatically. An interrupt will be generated if LBMI bit is set to "0" 1 = Load buffer0 is full. 0 = Load buffer0 is not full. Write '1' to clear. (write '0' leaves bit unchanged)
25 LBFL0	Load buffer 0 - full status bit. This bit indicates that log buffer registers are full. The bit is set to 1 automatically. An interrupt will be generated if LBMI bit is set to "0" 1 = Load buffer1 is full. 0 = Load buffer1 is not full. Write '1' to clear. (write '0' leaves bit unchanged)
24 DVFIS	DVFS Interrupt select. These bits define destination of DVFS interrupts. 1 = MCU interrupt will be generated for DVFS events. 0 = SDMA interrupt will be generated for DVFS events.

Table 24-14. DVFSCNTR Register Field Descriptions (continued)

Field	Description
23 PIRQS	PIRQS - Pattern IRQ Source * write '1' to clear. Writing '1' will clear interrupt if interrupt was from pattern 1 = DVFS IRQ source was from pattern 0 = DVFS IRQ source was not from pattern
22 FSVAIM	DVFS Frequency adjustment interrupt mask. This bit masks the DVFS frequency adjustment interrupt. FSVAI status bits are still asserted in relevant cases. 1 = interrupt is masked. 0 = interrupt is enabled.
21–20 FSVAI[1:0]	**FSVAI** DVFS Frequency adjustment interrupt. These status bits indicate that the system frequency should be changed. 00 = no interrupt 01 = frequency should be increased. Low priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency). 10 = frequency should be decreased. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MINF= 1 (lowest frequency). 11 = frequency should be increased immediately. High priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency).
19 WFIM	DVFS Wait for Interrupt mask bit 0 - Wait for interrupt not masked 1 - Wait for interrupt masked.
18 MAXF	Maximum frequency reached. Interrupt will not be created in maximum frequency reached and frequency increase required. 1 - max frequency reached 0 - max frequency not reached
17 MINF	Minimum frequency reached. Interrupt will not be created in minimum frequency reached and frequency decrease required. 1 - min frequency reached 0 - min frequency not reached
16–11 DIV_RATIO	DIV_RATIO—Divider value. Divider divides the input ARM clock, following Table 24-16.
10	Reserved
9 PFUE	PFUE - Period Frequency Update Enable 1 - enabled 0 - disabled
8-6 PFUS	PFUS - Periodic Frequency Update Status 000 - no update 100 - DVFSPT0 period, previous finished(can be performance level decrease) 101 - DVFSPT1 period, previous finished(can be EMA-detected performance level) 110 - DVFSPT2 period, previous finished(can be performance level increase) 111 - DVFSPT3 period, previous finished (can be EMA-detected performance level)
5 LTBRSH	LTBRSH - Load Tracking Buffer Register Shift: 0 = values of [5:2] of the selected input are saving in Load Tracking Buffer 1 = values of [4:1] of the selected input are saving in Load Tracking Buffer

Table 24-14. DVFSCNTR Register Field Descriptions (continued)

Field	Description
4–3 LTBRSR	LTBRSR - Load Tracking Buffer Register Source: 00 = pre_ld_add 01 = ld_add 10 = after ema 11 = reserved
2–1	reserved
0 DVFEN	DVFEN DVFS enable. This bit enables the DVFS block. 1 = DVFS enabled. 0 = DVFS disabled. NOTE: Between disable and enable there has to be at least 3 cycles of div_3_clk.

Table 24-15. DIV3CK Division

DIV3CK Setting	Dividing Ratio	sum_3 Passing Bits	div_1_clk Cumulative Divider
00	1	4-0	1*512=512
001	4	6-2	4*512=2048
010	16	8-4	16*512=8192
011	64	10-6	64*512=32768
100	256	12-8	256*512=131072
101	1024	16-10	1024*512=524288

Table 24-16. Preliminary Divider definition

DIV_RATIO value	ARM clk division ratio
000000	1
000001	2
000010	3
...	...

24.2.3.10 DVFSLTR0_0 Register

Figure 24-12 shows the DVFSLTR0_0 register, and Table 24-17 describes its register fields.

Address 0xBASE_0024 (DVFSLTR0_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS0_7				LTS0_6				LTS0_5				LTS0_4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_3				LTS0_2				LTS0_1				LTS0_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

7

Figure 24-12. DVFSLTR0_0 Register

Table 24-17. DVFLTR0_0 Register Field Descriptions

Field	Description
LTS0_7	Load Tracking Sample 7
LTS0_6	Load Tracking Sample 6
LTS0_5	Load Tracking Sample 5
LTS0_4	Load Tracking Sample 4
LTS0_3	Load Tracking Sample 3
LTS0_2	Load Tracking Sample 2
LTS0_1	Load Tracking Sample 1
LTS0_0	Load Tracking Sample 0

24.2.3.11 DVFSLTR0_1 Register

Figure 24-13 shows the DVFSLTR0_1 register, and Table 24-18 describes its register fields.

Address 0xBASE_0028 (DVFSLTR0_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS0_15				LTS0_14				LTS0_13				LTS0_12			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_11				LTS0_10				LTS0_9				LTS0_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

8

Figure 24-13. DVFSLTR0_1 Register

Table 24-18. DVFLTR0_1 Register Field Descriptions

Field	Description
LTS0_15	Load Tracking Sample 15
LTS0_14	Load Tracking Sample 14
LTS0_13	Load Tracking Sample 13
LTS0_12	Load Tracking Sample 12
LTS0_11	Load Tracking Sample 11
LTS0_10	Load Tracking Sample 10
LTS0_9	Load Tracking Sample 9
LTS0_8	Load Tracking Sample 8

24.2.3.12 DVFSLTR1_0 Register

Figure 24-14 shows the DVFSLTR1_0 register, and Table 24-19 describes its register fields.

Address 0xBASE_002C (DVFSLTR1_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS1_7				LTS1_6				LTS1_5				LTS1_4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS1_3				LTS1_2				LTS1_1				LTS1_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

9

Figure 24-14. DVFSLTR1_0 Register

Table 24-19. DVFLTR1_0 Register Field Descriptions

Field	Description
LTS1_7	Load Tracking Sample 7
LTS1_6	Load Tracking Sample 6
LTS1_5	Load Tracking Sample 5
LTS1_4	Load Tracking Sample 4
LTS1_3	Load Tracking Sample 3
LTS1_2	Load Tracking Sample 2
LTS1_1	Load Tracking Sample 1
LTS1_0	Load Tracking Sample 0

24.2.3.13 DVFSLTR1_1 Register

Figure 24-15 shows the DVFSLTR0_1 register.

Address 0xBASE_0030 (DVFSLTR1_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS1_15				LTS1_14				LTS1_13				LTS1_12			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS1_11				LTS1_10				LTS1_9				LTS1_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

10

Figure 24-15. DVFSLTR1_1 Register

24.2.3.14 DVFSPT0 Register

Figure 24-16 shows the DVFSPT0 register, and Table 24-20 describes its register fields.

Address 0xBASE_0034 (DVFSPT0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R															PT0A		FPTN0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Figure 24-16. DVFSPT0 Register

Table 24-20. DVFPT0 Register Field Descriptions

Field	Description
31–18	Reserved
17 PT0A	PT0A—Pattern 0 currently active (read-only) 1 Active 0 Non-active
16–0 FPTN0	FPTN0—Frequency pattern 0 counter During period of this counter the frequency will be reduced from the EMA-detected level.

24.2.3.15 DVFSPT1 Register

Figure 24-17 shows the DVFSPT1 register, and Table 24-21 describes its register fields.

Address 0xBASE_0038 (DVFSPT0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R															PT1A		FPTN1															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Figure 24-17. DVFSPT1 Register

Table 24-21. DVFSPT1 Register Field Descriptions

Field	Description
31–18	Reserved
17 PT1A	PT1A - Pattern 1 currently active (read-only) 1 Active 0 Non-active
16–0 FPTN1	FPTN1 - Frequency pattern 1 counter During period of this counter the frequency will be set to the EMA-detected level.

24.2.3.16 DVFSPT2 Register

Figure 24-18 shows the DVFSPT2 register, and Table 24-22 describes its register fields.

Address 0xBASE_003C (DVFSPT0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	P2THR														PT2A		FPTN2															
W																																
Reset	011111						0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Figure 24-18. DVFSPT2 Register

Table 24-22. DVFSPT2 Register Field Descriptions

Field	Description
31–26 P2THR	P2THR - Pattern 2 Threshold Threshold of current DVFS load (after EMA), for generating interrupts with PFUS indicators 110, 111. If the current performance is greater than the P2THR value, the interrupts will be generated. Otherwise, pattern delay will be counted, but without interrupt generation.
25–18	Reserved

Table 24-22. DVFSPT2 Register Field Descriptions (continued)

Field	Description
17 PT2A	PT2A - Pattern 2 currently active (read-only) 1 Active 0 Non-active
16–0 FPTN2	FPTN2 - Frequency pattern 2 counter During period of this counter the frequency will be increased to higher, than detected by the EMA-detected level.

24.2.3.17 DVFSPT3 Register

Figure 24-19 shows the DVFSPT3 register, and Table 24-23 describes its register fields.

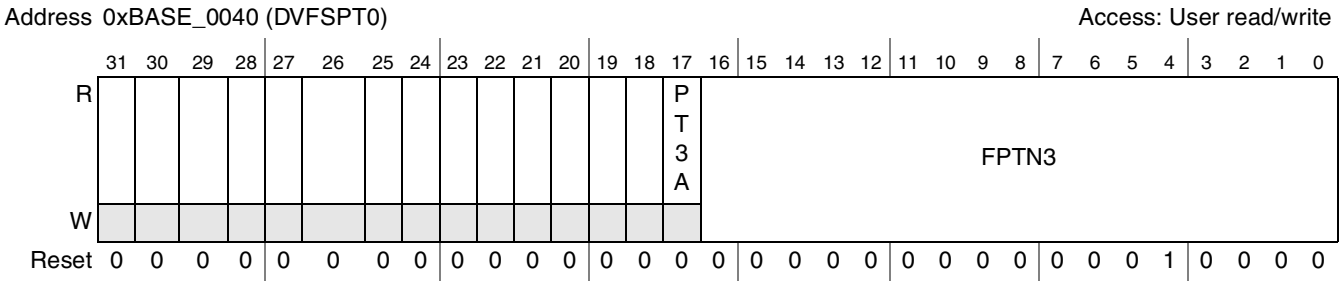


Figure 24-19. DVFSPT3 Register

Table 24-23. DVFSPT3 Register Field Descriptions

Field	Description
31–17	Reserved
18 PT3A	PT3A—Pattern 3 currently active (read-only) 1 Aactive 0 Non-active
16–0 FPTN3	FPTN3—Frequency pattern 3 counter During period of this counter the frequency will be set to the EMA-detected level.

24.3 Functional Description of DVFS Core Load Tracking

The DVFS load tracking block includes the following features:

Configurable include/exclude of input signals:

- ARM standby signal (idle/non-idle)
- 16 general purpose load tracking signals
- Configurable weight and (level-sensitive) of GeP signals.
- Configurable generated clocks and averaging time slicing (respond time).
- Configurable panic mode respond logic (for frequency up).
- Programmable buffer for last 4,8,12, or 16 load tracking samples. Two Load Tracking Buffers are working in ping-pong mode with two buffer full signals: LBFL1, LBFL0.

24.4 Component Blocks Description

This section provides descriptions and block diagrams for the component blocks.

24.4.1 dvfs_stdb_smpl block

This block samples the ccm_arm_stdb signal (ARM11 STANDBYWFI signal—idle state indicating) according to the edge of the ipg_clk_arm (ARM11 system clock) signal.

Figure 24-20 shows the dvfs_stdb_smpl block diagram.

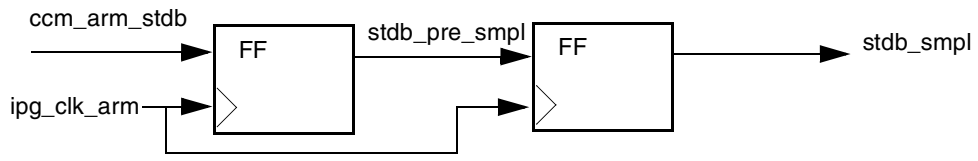


Figure 24-20. dvfs_stdb_smpl Block Diagram

This block synchronizes the ccm_arm_stdb signal with the ipg_clk_arm clock. The two signals enter the Flip-Flops (twice), and by doing so, are synchronized. The resulting synchronized signal is stdb_smpl.

24.4.2 dvfs_sig_wt block

The purpose of this block is to sample the 16 GeP (general purpose) load signals, multiply each one of them by the appropriate weight and sum products. The sampling is done by the slow clock div_3_clk.

These signals have the only one option of detection: level sampling. Figure 24-21 shows the block diagram.

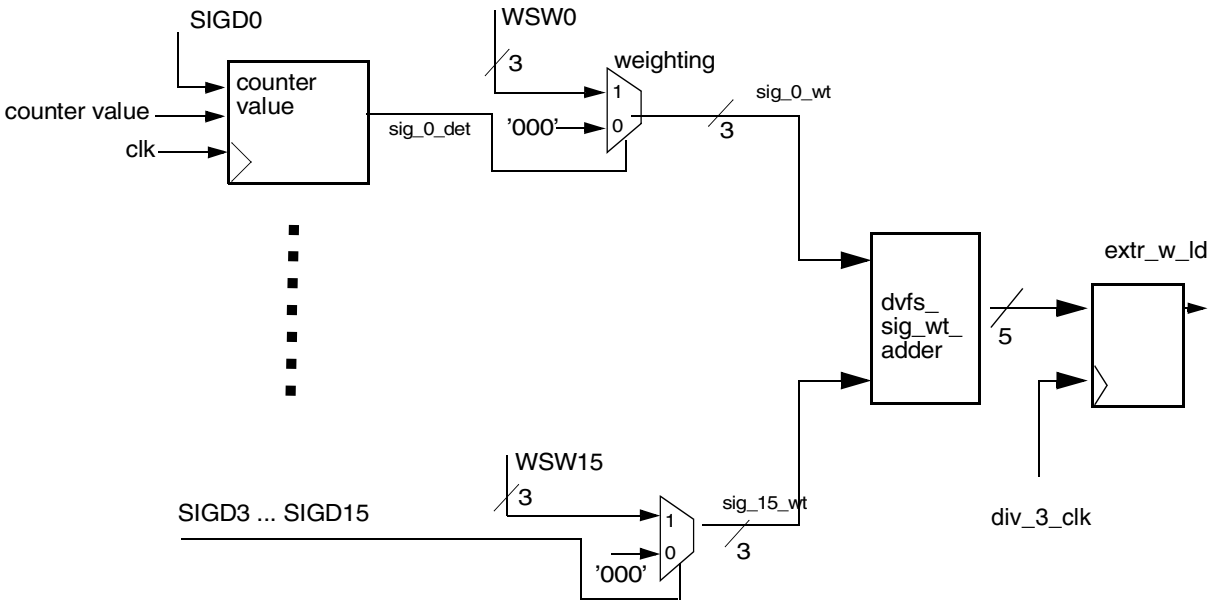


Figure 24-21. dvfs_sig_wt Block Diagram

There are two paths for general purpose bits weight.

24.4.3 dvfs_pre_avg block

The purpose of the dvfs_pre_avg block is to perform simple, non-overlapping averaging, reducing the sampling clock frequency and providing a level-based average index of the tracked CPU load.

External signals:

- stdb_sampl - (input) sampled ARM11 standby signal
- ipg_clk_arm - (input) ARM11 system clk
- pre_avg_ld[4..0] - (output) averaged load
- div_3_clk - (ouput) clock, generated (reduced) from div_2_clk
- div_2_clk - (output) clock, generated (reduced) from div_1_clk
- div_1_clk - (output) clock, generated (reduced) from ipg_clk_arm

In the current implementation, the averaging is performed in three stages:

1. The first stage !IDLE is sampled by divided sys_clock.

2. The second stage (counter2) performs an averaging operation with constant parameters. The counter is an nine (9) bit adder whose output is sampled by the div_2_clk clock. Five (5) highest bits of the sum_2 signal are passed to counter3 (equal to shift right operation).
3. The third (last) stage (counter3) performs an averaging operation with configurable parameters, set by the DIV3CK. The counter3 cell is a fifteen(15) bit adder with output sampled by div_3_clk. The Pre_avg_shifter_5 cell passes configurable bits from the sum_3 signal (see table1_3).

The output 5 bits avg_load signal provides a result with a tolerance of ~3%. (supported values range is 0–0.97)

Figure 24-22 shows the block diagram.

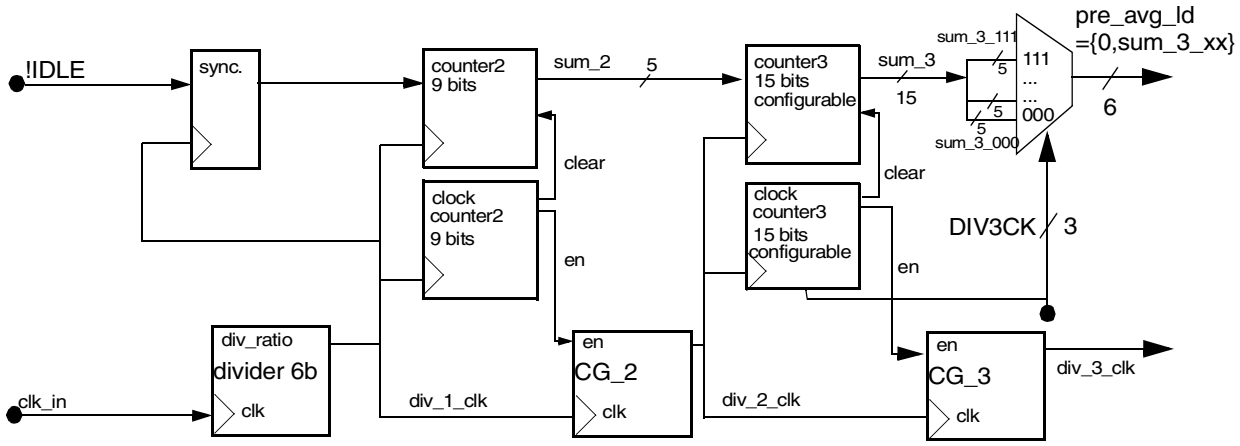


Figure 24-22. dvfs_pre_avg Block Diagram

Figure 24-23 shows the clocks.

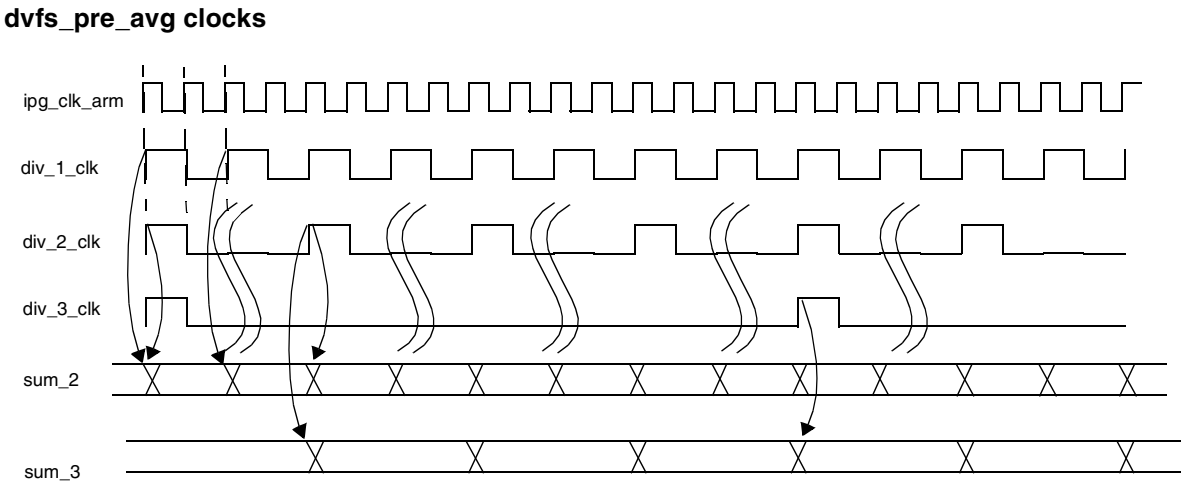


Figure 24-23. dvfs_pre_avg Clocks

The internal clocks in dvfs_avg_pre block are generated from ipg_clk_arm. See Table 24-24–Table 24-26 for details.

Table 24-24. dvfs_pre_avg Signals and its Values

Signal	Binary Max Value	Binary Min Value	Decimal Max Value	Decimal Min Value
sum_2	11111	0000	31	0
sum_3	111'1100'0000'0000	000'0000'0000'0000	31744	0
pre_avg_load	11111	0000	31	0

Table 24-25. dvfs_pre_avg Generated Clocks

Clock Name	Generated From	Ratio To Source	Ratio To lpg_clk_arm	Max Clk Freq.	Note
ipg_clk_arm	N/A	N/A	1	66MHz	source clock
div_1_clk	ipg_clk_arm	configurable	configurable	66MHz	configurable
div_2_clk (gated)	div_1_clk	512	configurable	128KHz	none
div_3_clk (gated)	div_2_clk	configurable	configurable	128KHz	configurable

Table 24-26. div_3_clk Configurable Averaging

DIV3CK Setting	Dividing Ratio	sum_3 Passing Bits	div_1_clk Cumulative Divider
00	1	4–0	$1 \times 512 = 512$
001	4	6–2	$4 \times 512 = 2048$
010	16	8–4	$16 \times 512 = 8192$
011	64	10–6	$64 \times 512 = 32768$

Table 24-26. div_3_clk Configurable Averaging (continued)

DIV3CK Setting	Dividing Ratio	sum_3 Passing Bits	div_1_clk Cumulative Divider
100	256	12–8	256 × 512 = 131072
101	1024	16–10	1024 × 512 = 524288

24.4.4 dvfs_ld_add Block

The dvfs_ld_add block sums the CPU load, tracked by idle/non-idle signal and the load, detected from the additional load signals, weighted by signal_weighting block.

The adder should perform the following operation:

extr_w_ld[4..0] + pre_avg_ld[4..0]

The input signals of five (5) bits produce output signal of six (6) bits, providing 3% resolution in the range of 0–1.97 of load indication. (1 is equal to 100% load tracking with no additional signals include).

The output ld_add[5..0] is sampled by div_3_clk signal.

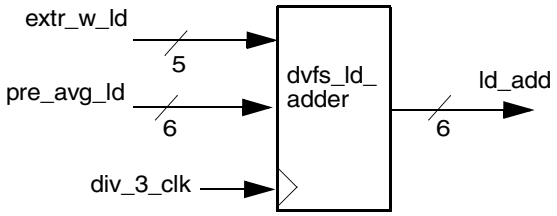


Figure 24-24. dvfs_ld_add Block Diagram

24.4.5 dvfs_ema_avg Block

The purpose of dvfs_ema_avg (EMA—Exponential Moving Average) block is to calculate an exponential moving average of the tracked CPU load. The parameters of EMA are defined by EMAC bits inside LTR2, bits ema_conf. These parameters set how many samples of simple average will be taken into account.

The EMA formula is: $EMA(i) = a \times X(i) + (1 - a) \times EMA(i - 1)$

where:

$a = 2 \div (N+1)$

N is the number of samples taken into account.

X(i) = current input sample

EMA(i – 1) = previous value of EMA

By setting the value of "a" , the behavior of EMA is defined.

In Table 24-27, the parameter "a" is listed relatively to the amount of the X(i) samples taken into account ("N") in the equation above: (the resolution of the digital multiplier is limited, hence the lowest values of the "a" can be linked to a range of included samples in EMA instead of single number).

In Table 24-27, there is also the amount of cycles in which the lower frequency is masked from the moment that DVFS is enabled (and not between frequency switches), so that enough information about the history can be gained before the frequency is lowered.

Table 24-27. EMA settings

Samples Included In Ema Calculation (N)	Number Of Cycles While Lower Frequency Interrupt Is Masked	"A" Value (Decimal)	"A" Value, Adjusted By Binary Resolution	"A" Value (Binary)								
----------		----------	----------	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8
1	6	1.000	1.000	1	0	0	0	0	0	0	0	0
2	11	0.667	0.668	0	1	0	1	0	1	0	1	1
3	11	0.500	0.500	0	1	0	0	0	0	0	0	0
4	22	0.400	0.398	0	0	1	1	0	0	1	1	0
5	22	0.333	0.332	0	0	1	0	1	0	1	0	1
6	22	0.286	0.285	0	0	1	0	0	1	0	0	1
7	22	0.250	0.250	0	0	1	0	0	0	0	0	0
8	43	0.222	0.223	0	0	0	1	1	1	0	0	1
9	43	0.200	0.199	0	0	0	1	1	0	0	1	1
10	43	0.182	0.180	0	0	0	1	0	1	1	1	0
11	43	0.167	0.168	0	0	0	1	0	1	0	1	1
12	43	0.154	0.152	0	0	0	1	0	0	1	1	1
13	43	0.143	0.141	0	0	0	1	0	0	1	0	1
14	43	0.133	0.133	0	0	0	1	0	0	0	1	0
15	43	0.125	0.125	0	0	0	1	0	0	0	0	0
16	86	0.118	0.117	0	0	0	0	1	1	1	1	0
17	86	0.111	0.109	0	0	0	0	1	1	1	0	0
18	86	0.105	0.105	0	0	0	0	1	1	0	1	1
19	86	0.100	0.102	0	0	0	0	1	1	0	1	0
20	86	0.095	0.094	0	0	0	0	1	1	0	0	0
21	86	0.091	0.090	0	0	0	0	1	0	1	1	1
22	86	0.087	0.086	0	0	0	0	1	0	1	1	0
23	86	0.083	0.082	0	0	0	0	1	0	1	0	1
25	86	0.077	0.078	0	0	0	0	1	0	1	0	0
26	86	0.074	0.074	0	0	0	0	1	0	0	1	1
28	86	0.069	0.070	0	0	0	0	1	0	0	1	0
30	86	0.065	0.066	0	0	0	0	1	0	0	0	1
31	86	0.063	0.063	0	0	0	0	1	0	0	0	0
33	173	0.059	0.059	0	0	0	0	0	1	1	1	1
35	173	0.056	0.055	0	0	0	0	0	1	1	1	0
38	173	0.051	0.051	0	0	0	0	0	1	1	0	1

Table 24-27. EMA settings

Samples Included In Ema Calculation (N)	Number Of Cycles While Lower Frequency Interrupt Is Masked	"A" Value (Decimal)	"A" Value, Adjusted By Binary Resolution	"A" Value (Binary)								
42	173	0.047	0.047	0	0	0	0	0	1	1	0	0
45	173	0.043	0.043	0	0	0	0	0	1	0	1	1
50	173	0.039	0.039	0	0	0	0	0	1	0	1	0
56	173	0.035	0.035	0	0	0	0	0	1	0	0	1
~64	173	0.031	0.031	0	0	0	0	0	1	0	0	0
~74	256	0.027	0.027	0	0	0	0	0	0	1	1	1
~86	256	0.023	0.023	0	0	0	0	0	0	1	1	0
~100	256	0.020	0.020	0	0	0	0	0	0	1	0	1
~128	256	0.016	0.016	0	0	0	0	0	0	1	0	0
~170	512	0.012	0.012	0	0	0	0	0	0	0	1	1
~260	512	0.008	0.008	0	0	0	0	0	0	0	1	0
~500	512	0.004	0.004	0	0	0	0	0	0	0	0	1
N/A		0.000	0.000	0	0	0	0	0	0	0	0	0

dvfs_ema_avg block diagram

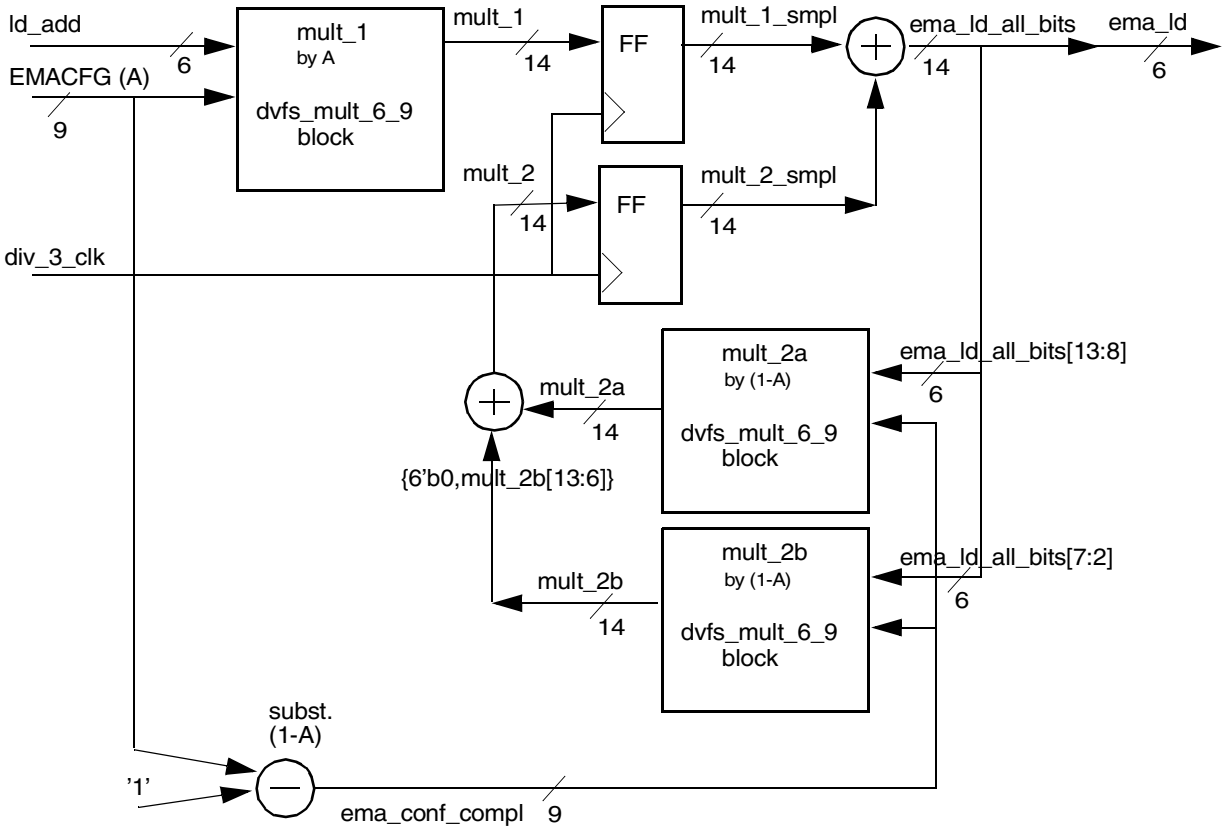


Figure 24-25. dvfs_ema_avg Block Diagram

The EMA block inputs are as follows:

- ld_add[5..0]—load level
- EMACFG[8..0]—"a" parameter of EMA algorithm
- div_2_clk—fast clock does not see this in the diagram
- div_3_clk—slow clock

The EMA block outputs the following:

- ema_ld[5..0] - result of EMA algorithm (by div_3_clk)

The mult_a block provides multiplying between comp_load (6 bits) and EMACFG (9 bits). For the multiply operation, a faster clock (for internal sum) is required- this is provided by div_2_clk. Div_2_clk is faster than the div_3_clk by a factor of at least 16, which is enough for 6×9 or 9×6 operations. Only the highest 6 bits are taken from the result of mult_a.

The mult_b block operates in a similar manner.

The adder block's output is sampled with div_3_clk clock.

24.4.6 dvfs_thres_cmp block

The dvfs_thres_cmp block compares the CPU load value to programmable threshold levels. The comparators as shown in Figure 24-26 are used:

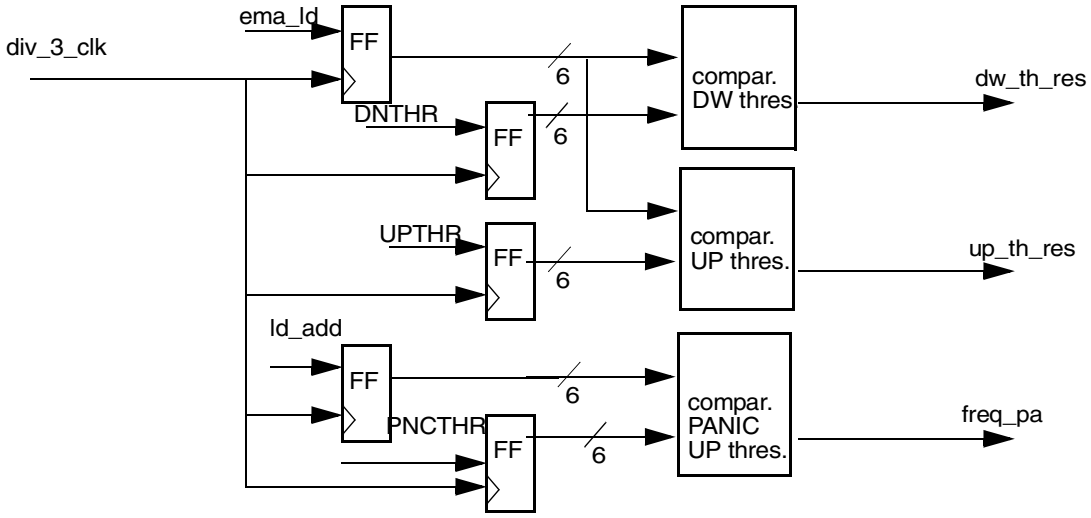


Figure 24-26. dvfs_tresh_cmp Diagram

The block is composed of three comparators:

- compar_dw_thres comparator, which compares between ema_ld signal (output of EMA block) and DNTHR signal (taken from config register, bits DNTHR).
- compar_up_thres comparator, which compares between ema_ld signal (output of EMA block) and UPTHR signal (taken from config register, bits UPTHR).
- compar_panic_up_thres comparator, which compares between ld_add signal (output of load_adder block) and PNCTHR signal (taken from config register, bits PNCTHR).

NOTE

The current implementation of this block enables step-by-step down frequency change. For more advanced option, the number of the DW threshold comparators should be increased (up to three for four-level DVFS).

24.4.7 dvfs_thresh_count block

The purpose of the dvfs_thres_count block is to count consecutive threshold overcomes of dw_th_res and up_th_res (outputs of threshold_comp block). If any of the counters (see Figure 24-27 below) receives a null (zero) input synchronous with the clk3 signal, the counter is reset.

If the counter reaches a user defined value, its output is set to an active level. These counters are reset each time frequency scaling occurs or if the threshold overcomes are consecutive.

This block's output signals are saved in configuration register 0, bits [3,2].

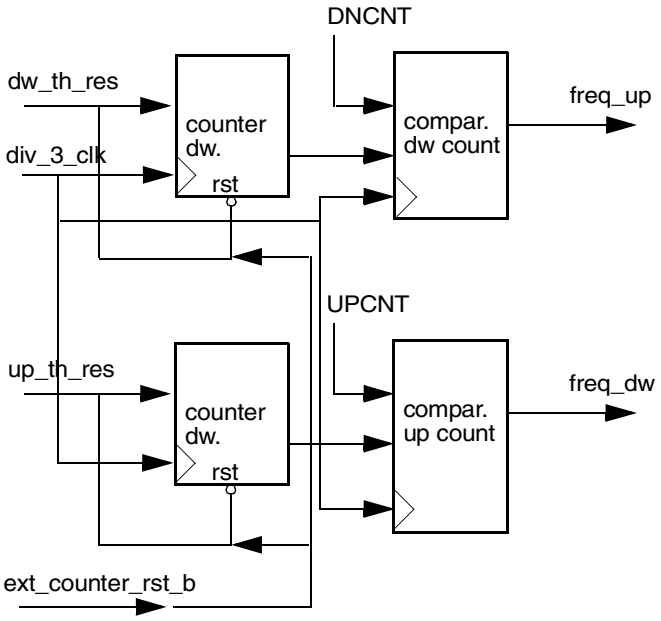


Figure 24-27. Thresh_counters Block Diagram

24.4.8 Load Tracking Buffer Register

The purpose of the load tracking buffer register is to save last 16 samples of the tracked load (before EMA operation). Hence, the 4 bits of ld_add signal (depending on the LTBRSH) are saved continuously, overwriting each time the latest sample. Each save is carried upon detecting an edge of the div_3_clk signal.

24.4.9 Frequency Pattern Generator

Frequency pattern generator is able to manage the frequency update requests periodically, additionally to main frequency update requests (if bit FPUE is set). The periodic requests are created following the DVFSPT0, DVFSPT1, DVFSPT2 and DVFSPT3 register values, as described in Figure 24-28.

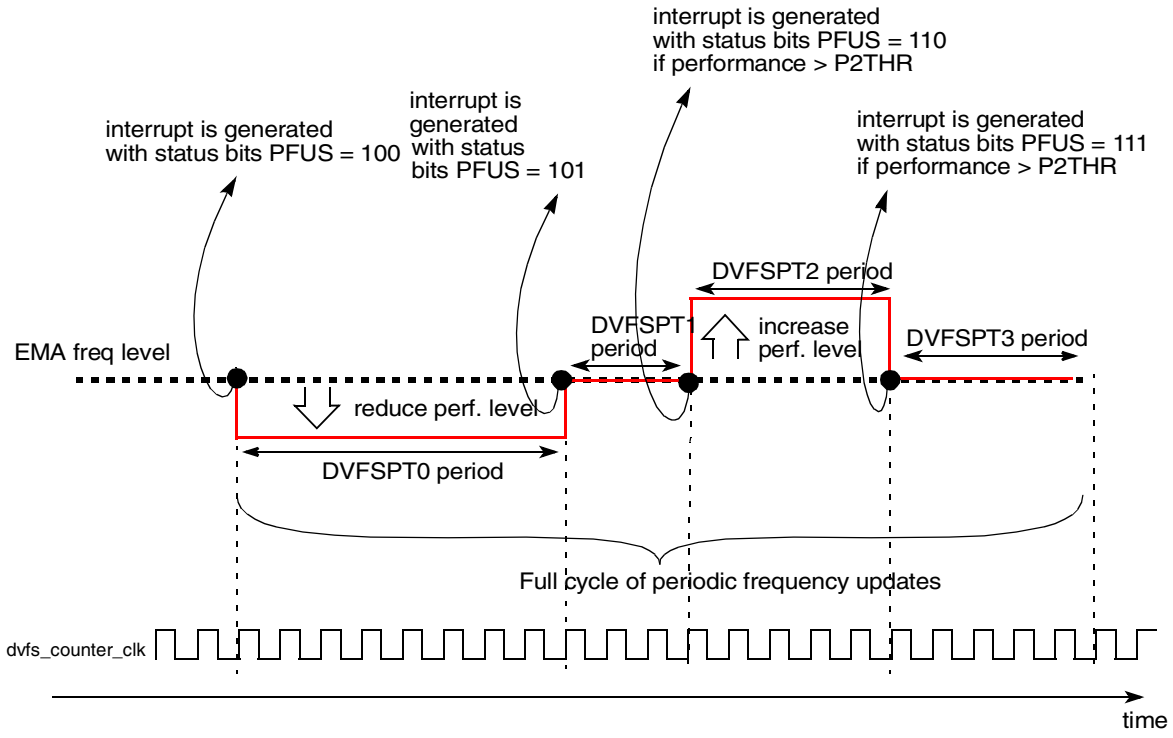


Figure 24-28. DVFS Periodic Frequency Update Requests

In case that one of the DVSPT0-DVFSPT3 period is set to “0”, such frequency update will be skipped.

The periodic frequency update status is reflected in PFUS bits (reduce/increase performance request is an example - the actual steps taken upon period expiration are defined by s/w routine).

Dvfs_counter_clk is ckih divided 64: 26MHz/64=0.40625MHz. The DVFSPT0, DVFSPT1, DVFSPT2 and DVFSPT3 counter are selected for 17 bits each to provide delay of 2^{18}-1=262143 counts, that are equal to 645ms. On the other hand, clk cycle of 0.40625MHz is ~2.46us, that is fast enough to provide high resolution for frequency management for tasks.

The DVFSPT2 and DVFSPT3 begin indication (interrupt) are conditional; only if the current performance is greater than P2THR bits at DVFSPT2 start, the interrupts will be created. Otherwise, the pattern delay will be counted, but without interrupt generation.

24.5 DVFS Output Event/interrupt Configuration

Event/Interrupt will be always high as long as LBFL is ’1’ and was not cleared by software. Unless DVFEV (always event) is asserted. Then the event/interruped will be toggled up and down every toggle of div_3_clk.

24.5.1 Interrupts

DVFS generates an interrupt that indicates that frequency and voltage update is needed. The user has to read the FSVAIM bits in order to know which change needs to be done.

24.6 Initialization Information

The user has to configure threshold values for load tracking and for counters and then enable the DVFS. When an interrupt is received, the user will change the frequency and the voltage in the interrupt handler.

NOTE

DVFS is a monitor that only provides an interrupt when counting exceeds the predefined value and does not actually send request to make a change a change of voltage and frequency. This can be done by the user in the interrupt routine or SDMA code by using GPC or CCM (relocking the PLL or changing the post dividers at the CCM).

Chapter 25
Dynamic Voltage Frequency Scaling for Peripherals (DVFS_PER)

25.1 Overview

Dynamic Voltage and Frequency Scaling (DVFS) enables the frequency of the system and the voltage of the power domain to be changed on the fly while all modules continue their normal operation on reduced frequency. The frequency of the clock domain is changed by dividing the clock source by DVFS divider. Serial clocks will remain the same.

The DVFS load tracking block allows hardware tracking on the system load and a generation of an interrupt when a frequency change is requested. The statistics for DVFS_PER are done on peripherals such as EMI (PER1), accelerators such as IPU, and so on.

NOTE

DVFS is a monitor that provides an interrupt when a count exceeds a predefined value. It does not request a change of voltage and frequency. This can be done by the user in the interrupt routine or SDMA code by using GPC or CCM (by locking the PLL or changing the post dividers at the CCM).

A diagram of the DVFS_PER module is shown in Figure 25-1.

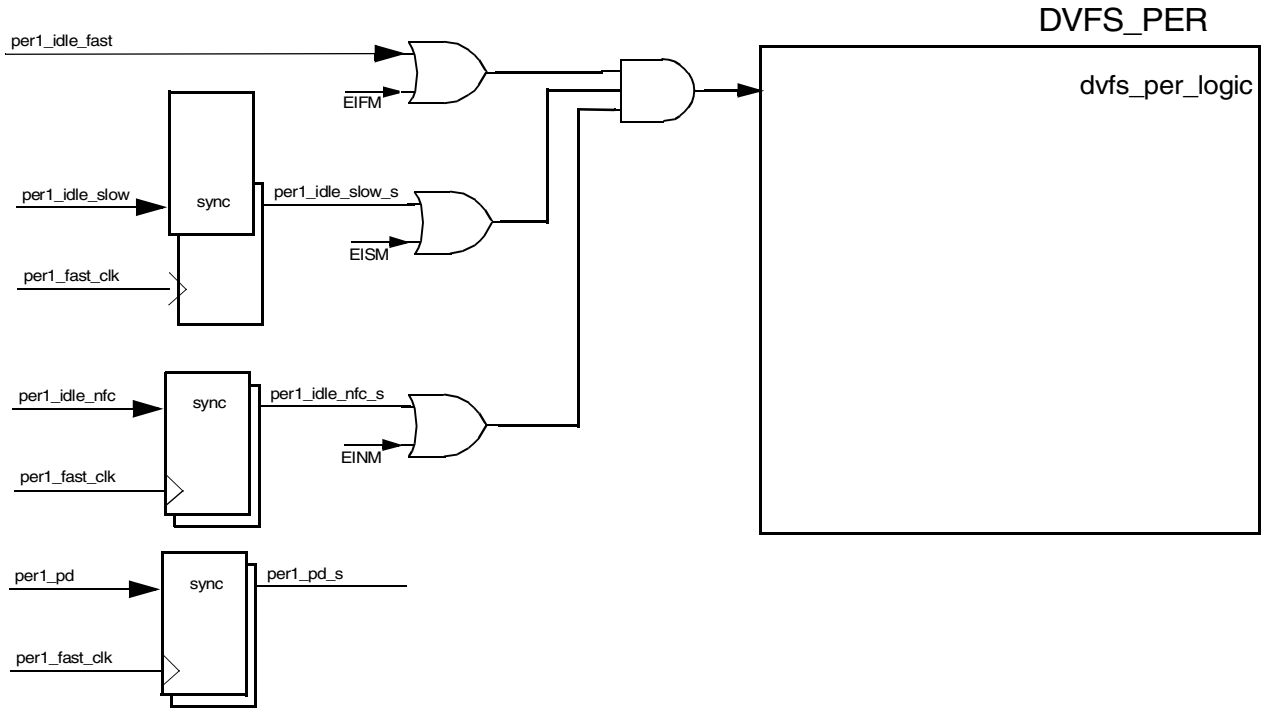


Figure 25-1. DVFS_PER Block Diagram

The DVFS_PER logic is shown in Figure 25-2.

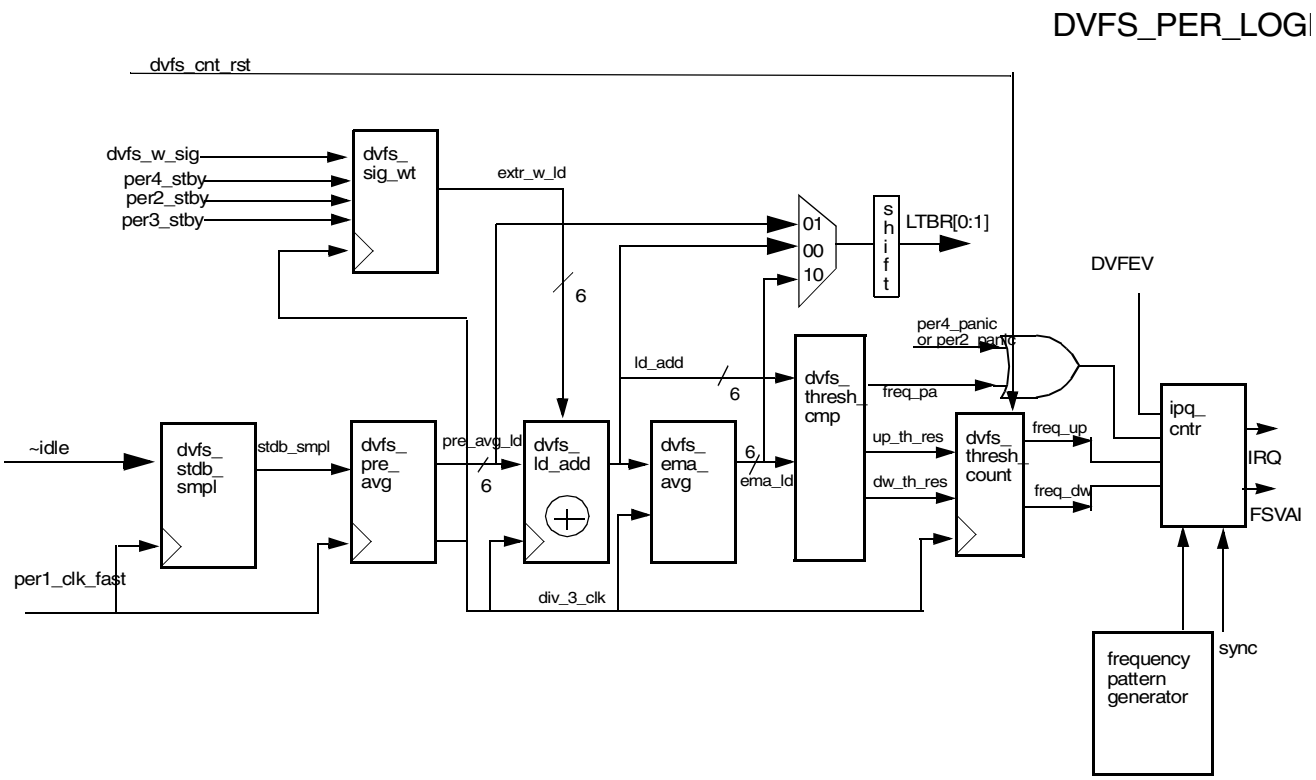


Figure 25-2. DVFS_PER_LOGIC Block Diagram

25.1.1 Features

The DVFS load tracking block includes the following features:

- Configurable include/exclude of input signals:
  - — Peripheral_1 (PER1) standby signal (idle/non-idle)
  - — 10 general purpose bits
  - — Peripheral_2 (PER2) standby
  - — Peripheral_3 (PER3) standby
  - — Peripheral_4 (PER4) standby
    - – Configurable weight of each bit.
- Configurable generated clocks and averaging time slicing (respond time).
- Configurable panic mode respond logic (for frequency up).
- Programmable buffer for last 4,8,12, or 16 load tracking samples based on value in LBCF in PMCR0. There is also a buffer full signal, LBFL.

25.2 Memory Map and Register Definition

This section provides a memory map, register figure key, and a register summary.

25.2.1 Memory Map

This DVFS_PER memory map is shown in Table 25-2.

Table 25-2. DVFS_PER Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
BASE_ADDR + 0004	LTR0—DVFS Load Tracking Register 0	R	0x0714_0006	25.2.3.1/25-7
BASE_ADDR + 0008	LTR1—DVFS Load Tracking Register 1	R	0x000F_D53E	25.2.3.2/25-8
BASE_ADDR + 000c	LTR2—DVFS Load Tracking Register 2	R	0x0000_0000	25.2.3.3/25-9
BASE_ADDR + 0010	LTR3—DVFS Load Tracking Register 3	R	0x0000_0000	25.2.3.4/25-10
BASE_ADDR + 0014	LTBR0	R/W	0x0000_0000	25.2.3.5/25-11
BASE_ADDR + 0018	LTBR1	R/W	0x0000_0000	25.2.3.6/25-12
BASE_ADDR + 001C	PMCR0	R/W	0x082C_0000	25.2.3.7/25-12
BASE_ADDR + 0020	PMCR1	R/W	0x0000_0000	25.2.3.8/25-14

25.2.2 Register Summary

The conventions in Figure 25-3 and Table 25-3 serve as a key for the register summary and individual register diagrams.

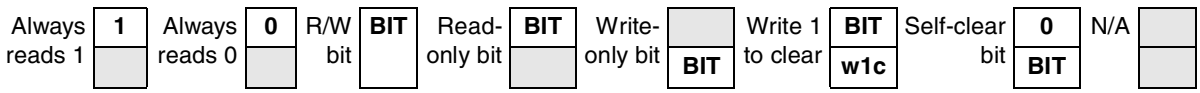


Figure 25-3. Key to Register Fields

Table 25-3 provides a key for register figures and tables and the register summary.

Table 25-3. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.

Table 25-3. Register Conventions (continued)

Convention	Description
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

The DVFS_PER registers are summarized in Table 25-4.

Table 25-4. DVFS_PER Registers Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR + 0004 LTR0	R	SIG D15	SIG D14	SIG D13	0	UPTHR						DNTHR					
	W																
	R	SIG D12	SIG D11	SIG D10	SIG D9	SIG D8	SIG D7	SIG D6	SIG D5	SIG D4	SIG D3	SIG D2	SIG D1	SIG D0	DIV3CK		
	W																
BASE_ADDR + 0008 LTR1	R	DIV_RATIO[5:0]						0	0	LTB RSH	LTB RSR	DNCNT[7:2]					
	W																
	R	DNCNT[1:0]		UPCNT[7:0]								PNCTHR[5:0]					
	W																
BASE_ADDR + 000C LTR2	R	WSW15		WSW14		WSW13			WSW12			WSW11			WSW10		
	W																
	R	WSW10		WSW9		0	0	EMAC[8:0]									
	W																

Table 25-4. DVFS_PER Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR + 0010 LTR3	R	WSW8			WSW7			WSW6			WSW5			WSW4			WSW3[2]
	W																
	R	WSW3[1:0]		WSW2			WSW1			WSW0							
	W																
BASE_ADDR + 0014 LTBR0	R	LTS0_7				LTS0_6				LTS0_5				LTS0_4			
	W																
	R	LTS0_3				LTS0_2				LTS0_1				LTS0_0			
	W																
BASE_ADDR + 0018 LTBR1	R	LTS0_15				LTS0_14				LTS0_13				LTS0_12			
	W																
	R	LTS0_11				LTS0_10				LTS0_9				LTS0_8			
	W																
BASE_ADDR + 001C PMCR0	R					UDCS				DVFEV	DVFIS	LBMI	LBFL	LBCF			
	W																
	R	FSVAIM	FSVAI				WFIM						DVFEN				
	W																
BASE_ADDR + 0020 PMCR1	R												P1INM	P1ISM	P1IFM	P4PM	P2PM
	W																
	R	DVGP[15:0]															
	W																

25.2.3 Register Descriptions

This section provides register figures and register field descriptions for each of the DVFS-P registers.

25.2.3.1 LTR0 register

The LTR0 register is shown in Figure 25-4.

Address 0xBASE_0004 (LTR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SIGD 15	SIGD 14	SIGD 13	0	UPTHR						DNTHR					
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SIGD 12	SIGD 11	SIGD 10	SIGD 9	SIGD 8	SIGD 7	SIGD 6	SIGD 5	SIGD 4	SIGD 3	SIGD 2	SIGD 1	SIGD 0	DIV3CK		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Figure 25-4. LTR0 Register

The LTR0 fields are described in Table 25-10.

Table 25-5. LTR0 Register Field Descriptions

Field	Description
31 SIGD15	Detect way of general signal #15 (per2_not_idle): 1= edge, 0 = level
30 SIGD14	Detect way of general signal #14 (per2_not_idle): 1 = edge, 0 = level
29 SIGD13	Detect way of general signal #13 (per3_not_idle): 1 = edge, 0 = level
28	Reserved
27–22 UPTHR	Upper threshold for load tracking
21–16 DWTHR	Down threshold for load tracking
15 SIGD12	Detect way of general signal #12 (per3_not_idle): 1 = edge, 0 = level
14 SIGD11	Detect way of general signal #11 (per4_not_idle): 1 = edge, 0 = level
13 SIGD10	Detect way of general signal #10 (per4_not_idle): 1 = edge, 0 = level

Table 25-5. LTR0 Register Field Descriptions (continued)

Field	Description
12 SIGD9 ... 3 SIGD0	Detect way of general signal #9 ... #0(register): 1=edge, 0=level
2–0 DIV3CK	DIV3CK - div_3_clk division ratio inside the DVFS module. See Table 25-10 for the significance of different field values.

DIV3CK setting	dividing ratio	sum_3 passing bits	div_1_clk cumulative divider
00	1	4-0	1*512=512
001	4	6-2	4*512=2048
010	16	8-4	16*512=8192
011	64	10-6	64*512=32768
100	256	12-8	256*512=131072
101	1024	16-10	1024*512=524288

25.2.3.2 LTR1 Register

The LTR1 register fields are shown in Figure 25-5., and the fields are described in Table 25-11.

Address 0xBASE_0008 (LTR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DIV_RATIO[5:0]						0	0	LTBR SH	LTBR SR	DNCNT[7:2]					
Reset	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DNCNT[1:0]		UPCNT[7:0]						PNCTHR[5:0]							
Reset	1	1	0	1	0	1	0	1	0	0	1	1	1	1	1	0

11

Figure 25-5. LTR1 Register

Table 25-6. LTR1 Register Field Descriptions

Field	Description
31–26 DIV_RATIO	DIV_RATIO - Divider value. Divider divides the input ARM clock: 000000ARM clock division ratio is 1 000001ARM clock division ratio is 2 000010ARM clock division ration is 3 000011–111111Reserved
25–24 Reserved	Reserved.
23 LTBRSH	LTBR source shift: (source by ltbrsr bit)
22 LTBRSR	LTBR source signal 0 = pre_avg_l 1 = ld_add
21–4 DNCNT	Down counter threshold value
13–6 UPCNT	UP counter threshold value
5–0 PNCTHR	Panic threshold value

25.2.3.3 LTR2 Register

The LTR2 register fields are shown in Figure 25-6, and the fields are described in Table 25-7.

Address 0xBASE_000C (LTR2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WSW15				WSW14				WSW13				WSW12			WSW 10
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WSW10			WSW9					EMAC[8:0]							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

12

Figure 25-6. LTR2 Register

Table 25-7. LTR2 Register Field Descriptions

Field	Description
31–29 WSW15	General purpose load tracking signal weight dvfs_w_sig[15]
28–26 WSW14	General purpose load tracking signal weight dvfs_w_sig[14]

Table 25-7. LTR2 Register Field Descriptions (continued)

Field	Description
25–23 WSW13	General purpose load tracking signal weight dvfs_w_sig[13]
22–20 WSW12	General purpose load tracking signal weight dvfs_w_sig[12]
19–17 WSW11	General purpose load tracking signal weight dvfs_w_sig[11]
16–14 WSW10	General purpose load tracking signal weight dvfs_w_sig[10]
13–11 WSW9	General purpose load tracking signal weight dvfs_w_sig[9]
10–9	Reserved
8–0 EMAC	EMAC—EMA algorithm value

25.2.3.4 LTR3 Register

The LTR3 register fields are shown in Figure 25-7, and the fields are described in Table 25-8.

Address 0xBASE_0010 (LTR3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WSW8				WSW7				WSW6				WSW5			WSW 3[2]
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WSW3[1:0]			WSW2			WSW1		WSW0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13

Figure 25-7. LTR3 Register

Table 25-8. LTR3 Register Field Descriptions

Field	Description
31–29 WSW8	General purpose load tracking signal weight dvfs_w_sig[8]
28–26 WSW7	General purpose load tracking signal weight dvfs_w_sig[7]
25–23 WSW6	General purpose load tracking signal weight dvfs_w_sig[6]
22–20 WSW5	General purpose load tracking signal weight dvfs_w_sig[5]

Table 25-8. LTR3 Register Field Descriptions (continued)

Field	Description
19–17 WSW4	General purpose load tracking signal weight dvfs_w_sig[4]
16–14 WSW3	General purpose load tracking signal weight dvfs_w_sig[3]
13–11 WSW2	General purpose load tracking signal weight dvfs_w_sig[2]
10–8 WSW1	General purpose load tracking signal weight dvfs_w_sig[1]
7–5 WSW0	General purpose load tracking signal weight dvfs_w_sig[0]
4–0	Reserved

25.2.3.5 LTBR0 Register

The LTBR0 register fields are shown in Figure 25-8, and the fields are described in Table 25-9.

Address 0xBASE_0014 (LTBR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS0_7				LTS0_6				LTS0_5				LTS0_4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_3				LTS0_2				LTS0_1				LTS0_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

14

Figure 25-8. LTBR0 Register

Table 25-9. LTBR0 Register Field Descriptions

Field	Description
LTS0_7	Load Tracking Sample 7
LTS0_6	Load Tracking Sample 6
LTS0_5	Load Tracking Sample 5
LTS0_4	Load Tracking Sample 4
LTS0_3	Load Tracking Sample 3
LTS0_2	Load Tracking Sample 2
LTS0_1	Load Tracking Sample 1
LTS0_0	Load Tracking Sample 0

25.2.3.6 LTBR1 Register

The LBTR1 register fields are shown in Figure 25-9., and the fields are described in Table 25-10.

Address 0xBASE_0018 (LTBR1_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTS0_15				LTS0_14				LTS0_13				LTS0_12			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_11				LTS0_10				LTS0_9				LTS0_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15

Figure 25-9. LTBR1 Register

Table 25-10. LTBR1 Register Field Descriptions

Field	Description
LTS0_15	Load Tracking Sample 15
LTS0_14	Load Tracking Sample 14
LTS0_13	Load Tracking Sample 13
LTS0_12	Load Tracking Sample 12
LTS0_11	Load Tracking Sample 11
LTS0_10	Load Tracking Sample 10
LTS0_9	Load Tracking Sample 9
LTS0_8	Load Tracking Sample 8

25.2.3.7 PMCR0 register

The PMCR0 register fields are shown in Figure 25-10, and the fields are described in Table 25-11.

Address 0xBASE_001C (PMCR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R					UDCS				DVFEV	DVFIS	LBMI	LBFL	LBCF			
W																
Reset	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FSVAIM	FSVAI				WFIM						DVFEN				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16

Figure 25-10. PMCR0 Register

Table 25-11. PMCR0 Register Field Descriptions

Field	Description
31–28	Reserved
27 UDSC	Up-down scaling. This bit indicates the direction of current frequency scaling. 1 = Frequency is increased. 0 = Frequency is decreased.
26–24	Reserved
23 DVFEV	Always give a DVFS event. 0 - Do not always give event. 1- Always give event.
22 DVFIS	DVFS Interrupt select. These bits define destination of DVFS interrupts. 1 = MCU interrupt will be generated for DVFS events. 0 = SDMA interrupt will be generated for DVFS events.
21 LBMI	Load buffer full mask interrupt. This bit masks the generation of this interrupt. 1 = Load buffer full interrupt is masked. 0 = Load buffer full interrupt is enabled.
20 LBFL	Load buffer - full status bit. This bit indicates that log buffer registers are full. The bit is set to 1 automatically. An interrupt will be generated if LBMI bit is set to "0" 1 = Load buffer is full. 0 = Load buffer is not full.
19–18 LBCF	load tracking configuration: 00 = 4 samples, 01 = 8 samples, 10 = 12 samples, 11 = 16 samples
17–16	Reserved
15 FSVAIM	DVFS Frequency adjustment interrupt mask. This bit masks the DVFS frequency adjustment interrupt. FSVAI status bits are still asserted in relevant cases. 1 = interrupt is masked. 0 = interrupt is enabled.
14–13 FSVAI [1:0]	**FSVAI** DVFS Frequency adjustment interrupt. These status bits indicate that the system frequency should be changed. 00 = no interrupt 01 = frequency should be increased. Low priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency). 10 = frequency should be decreased. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MINF= 1 (lowest frequency). 11 = frequency should be increased immediately. High priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency).
12–11	reserved
10	WFIM - WFI ARM signal masking. 1 = WFI ARM is masked 0 = WFI ARM is not masked
9–5	Rreserved

Table 25-11. PMCR0 Register Field Descriptions (continued)

Field	Description
4 DVFEN	DVFEN DVFS enable. This bit enables the DVFS block. 1 = DVFS enabled. 0 = DVFS disabled. NOTE: Between disable and enable there has to be at least 3 cycles of div_3_clk.
3–0	reserved

25.2.3.8 PMCR1 register

Address 0xBASE_0020 (PMCR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												P1INM	P1ISM	P1IFM	P4PM	P2PM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DVGP[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

17

Figure 25-11. PMCR1 Register

Table 25-12. PMCR1 Register Field Descriptions

Field	Description
31–21	reserved
20 P1INM	PER1 Idle NFC Mask 1 - per1_idle_nfc is masked 0 - not masked
19 P1ISM	PER1 Idle Slow Mask 1 - per1_idle_slow is masked 0 - not masked
18 P1IFM	PER1 Idle Fast Mask 1 - per1_idle_fast is masked 0 - not masked
17 P4PM	PER4 panic mask 1= PER4 panic input is masked
16 P2PM	PER2 panic mask 1= PER2 panic input is masked
15–0 DVGP	General purpose bits, used for WSW

25.3 Functional Description of DVFS Core Load Tracking

The DVFS load tracking block includes the following features:

Configurable include/exclude of input signals:

- PER1 standby signal (idle/non-idle)
- 10 general purpose load tracking signals
- per2 idle
- per3 idle
- per4 idle
- per2 panic
- per4 panic
- Configurable weight and (level-sensitive) of GeP signals.
- Configurable generated clocks and averaging time slicing (respond time).
- Configurable panic mode respond logic (for frequency up).
- Programmable buffer for last 4, 8, 12, or 16 load tracking samples.

Chapter 26
Enhanced Configurable Serial Peripheral Interface (eCSPI)

26.1 Introduction

The eCSPI module allows rapid data communication with fewer software interrupts than conventional serial communications. The eCSPI module contains one 64 × 32 receive buffer (RXFIFO) and one 64 × 32 transmit buffer (TXFIFO). Figure 26-1 shows the eCSPI block diagram.

26.1.1 Overview

The eCSPI is equipped with data FIFOs and is a master/slave configurable serial peripheral interface module, capable of interfacing to both SPI master and slave devices. The eCSPI Ready ($\overline{\text{SPI\_RDY}}$) and Chip Select ($\overline{\text{SS}}$) control signals enable fast data communication with fewer software interrupts.

There are two identical eCSPI modules in the IC. This chapter describes the configuration and operation of the eCSPI. Information such as base addresses and features that are unique to the multiple devices is described as appropriate.

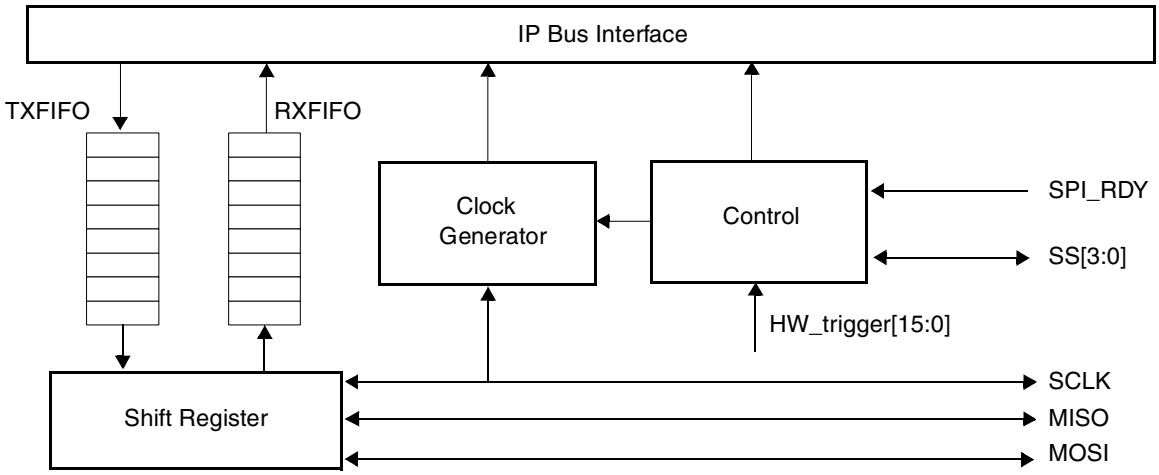


Figure 26-1. eCSPI Block Diagram

26.1.2 Features

The eCSPI is used for fast data communication with fewer software interrupts. It includes the following features:

- Four SPI channel with separate configuration bits and signals to support Multiple peripherals.
- Full-duplex synchronous serial interface

- Master/Slave configurable
- Support SPI clock up to 66MHz in both Master and Slave mode
- Transfer continuation function allows unlimited length data transfers
- 32-bit wide by 64-entry FIFO for both transmit and receive data
- 32-bit wide by 16-entry FIFO for HT message data
- Polarity and phase of the Chip Select ($\overline{SS}$) and SPI Clock (SCLK) are configurable
- DMA support
- Support hardware Trigger Mode (HT Mode).

26.1.3 Modes of Operation

eCSPI has three normal operating modes: master mode, slave mode and HT mode. It can be worked under low-power mode by enabling clock inputs to eCSPI.

eCSPI also provides some test register bits for debug purpose.

26.1.3.1 Normal Operating Modes

The normal operating modes are as follows:

- Master Mode

  When the eCSPI module is configured as a master, it uses a serial link to transfer data between the eCSPI and an external device. A chip-enable signal and a clock signal are used to transfer data between these two devices. If the external device is a transmit-only device, the eCSPI master's output port can be ignored and used for other purposes. In order to use the internal TXFIFO and RXFIFO, two auxiliary output signals, $\overline{SS}$ and $\overline{SPI\_RDY}$, are used for data transfer rate control. The user can also program the sample period control register to a fixed data transfer rate.
- Slave Mode

  When the eCSPI module is configured as a slave, the user can configure the eCSPI Control register to match the external SPI master's timing. In this configuration, $\overline{SS}$ becomes an input signal, and is used to control data transfers through the Shift register, as well as to load/store the data FIFO.
- HT Mode

  In this mode, all 16 message contents are stored in MSG Data FIFO inside the eCSPI which are associated with a set of input strobe signals. This mode is typically used to enable efficient PMIC control (via eCSPI) by GPC module, to support DVFS/DPTC with minimal CPU intervention.
  - Support up to 16 sets of 32-bit HT Message contents. And the length of HT Message contents can be programmed from 1 to 32.
  - Each HT Message content is associated with a Hardware Trigger input strobe signal and is triggered on rising edge of this signal.
  - HT Mode is valid with eCSPI Master Mode and only CS0 is available for HT mode.
  - Multiple simultaneous Hardware Triggers are not supported. If they appear, a single Hardware Trigger of highest priority is serviced. The priority of Hardware Triggers decreases with increase of their number, such as HT 0 has highest priority and HT 15 has lowest priority.

— If eCSPI transactions are requested simultaneously by CPU and Hardware Trigger, HT Mode has highest priority. If Hardware Trigger or CPU request arrive while the other one is serviced, it will be pending for the previous transaction completion.

26.1.3.2 Low Power Modes

eCSPI needs both ipg_clk and ipg_clk_per for its normal operations, so in low-power mode eCSPI works only if all these clock inputs are enabled.

26.1.3.3 Debug Mode

eCSPI provides the following test register bits for debug purposes.

- LBC bit in Test Register—provides a way to shift data out from TXFIFO and then loop back them into RXFIFO.
- CL bits in Test Register—provides a way to latch data in time when path delay is large.
- TXCNT bits in Test Register—provides a way to determine how many data are there in TXFIFO.
- RXCNT bits in Test Register—provides a way to determine how many data are there in RXFIFO.

26.2 External Signal Description

The following signals shown in Table 26-1 are used to control the serial peripheral interface.

Table 26-1. eCSPI Signal Properties

Name	Function	I/O	Reset	Pull-up
MOS	Master data out; slave data in	I/O	1’b0	—
MISO	Master data in; slave data out	I/O	1’b0	—
SCLK	SPI clock	I/O	1’b0	—
$\overline{SS}$[3:0]	Chip selects	I/O	4’b1111	Active
SPI_RDY	SPI data ready in Master mode	I	1’b1	Active
ipp_obe_mosi	Output enable for ipp_do_mosi, ipp_cspi_clk_out, ipp_do_ss_b	O	1’b0	—
ipp_obe_miso	Output enable for ipp_do_miso	O	1’b1	—

Table 26-2 provides a detailed description of all eCSPI external signals.

Table 26-2. eCSPI – Detailed External Signal Descriptions

Signal	I/O	Description	
MOSI	I/O	Master Out Slave In. In Master mode, this bidirectional signal is a TX output signal from the Data Shift register. In Slave mode, it is an RX input from external SPI device.	
		State Meaning	Asserted—Indicates a 1 is transmitted. Negated—Indicates a 0 is transmitted.
		Timing	Assertion—May occur at any edge of the SPI clocks according to configuration of eCSPI. Negation— May occur at any edge of the SPI clocks according to configuration of eCSPI.
MISO	I/O	Master In Slave Out – In Master mode, this bidirectional signal is an RX input signal to the Data Shift register. In Slave mode, it is a TX output to external SPI device.	
		State Meaning	Asserted—Indicates a 1 is transmitted. Negated—Indicates a 0 is transmitted.
		Timing	Assertion—May occur at any edge of the SPI clocks according to configuration of eCSPI. Negation— May occur at any edge of the SPI clocks according to configuration of eCSPI.
SCLK	I/O	SPI Clock – In Master mode, this bidirectional signal is a SPI clock output. In Slave mode, it is a SPI clock input.	
		State Meaning	Asserted—Clock is high. Negated—Clock is low.
		Timing	Assertion—May occur at any time. Negation— May occur at any time
$\overline{SS}$[3:0]	I/O	Chip selects – In Master mode, these bidirectional signals are outputs. In Slave mode, these are inputs. These signals are selected in/out by Chip Select bits in the eCSPI Control register.	
		State Meaning	Asserted—When SSPOL is set, device is selected, otherwise device is de-selected. Negated—When SSPOL is set, device is de-selected, otherwise device is selected.
		Timing	Assertion—May occur at any edge of the SPI clocks according to configuration of eCSPI. Negation— May occur at any edge of the SPI clocks according to configuration of eCSPI.
SPI_RDY	I	SPI Ready – It is an input from an external SPI slave device. This signal triggers the eCSPI to start a burst. Edge-trigger or level-trigger can be configured in the eCSPI Control register.	
		State Meaning	Asserted—External SPI device is not ready. Negated—External SPI device is ready.
		Timing	Assertion—May occur at any time. Negation— May occur at any time.

26.3 Memory Map and Register Definition

The eCSPI includes twenty-five 32-bit registers. Section 26.3.3, Register Descriptions,” provides detailed descriptions for the eCSPI registers.

26.3.1 Memory Map

There are two eCSPI modules in the i.MX51. Table 26-3 shows the eCSPI memory map for eCSPI1. The offsets for eCSPI2 are 0xBASE+0x000 through 0xBASE+0x040.

Table 26-3. eCSPI Memory Map[1]

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x000 (RXDATA)	RXFIFO DATA Register (RXDATA)	R	0x0000_0000	26.3.3.1/26-7
0xBASE+0x004 (TXDATA)	TXFIFO DATA Register (TXDATA)	W	0x0000_0000	26.3.3.2/26-8
0xBASE+0x008 (CONTROLREG)	Control Register (CONTROLEG)	R/W	0x0000_0000	26.3.3.3/26-9
0xBASE+0x00C (CONFIGREG)	Config Register (CONFIGREG)	R/W	0x0000_0000	26.3.3.4/26-11
0xBASE+0x010 (INTREG)	Interrupt Control Register (INTREG)	R/W	0x0000_0000	26.3.3.5/26-13
0xBASE+0x014 (DMAREG)	DMA Control Register (DMAREG)	R/W	0x0000_0000	26.3.3.6/26-14
0xBASE+0x018 (STATUSREG)	Status Register (STATREG)	R/W	0x0000_0003	26.3.3.7/26-16
0xBASE+0x01C (PERIODREG)	Sample Period Control Register (PERIODREG)	R/W	0x0000_0000	26.3.3.8/26-17
0xBASE+0x020 (TESTREG)	Test Control Register (TESTREG)	R/W	0x0000_0000	26.3.3.9/26-18
0xBASE+0x040 (MSGDATA)	Message Data Register (MSGDATA)	R/W	0x0000_0000	26.3.3.10/26-19

[1] See Chapter 2, "Memory Map," for the value of base address of eCSPI1 and eCSPI2.

26.3.2 Register Summary

The conventions in Figure 26-2 and Table 26-4 serve as a key for the register summary and individual register diagrams.

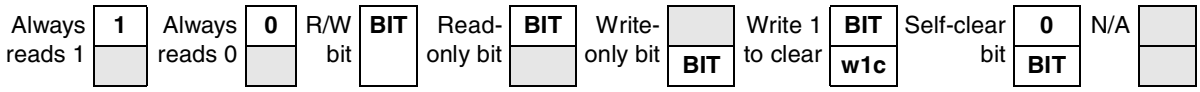


Figure 26-2. Key to Register Fields

Table 26-4 provides a key for register figures and tables and the register summary.

Table 26-4. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).

Table 26-4. Register Conventions (continued)

Convention	Description
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 26-5 shows the eCSPI register summary.

Table 26-5. eCSPI Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (RXDATA)	R	RXFIFO[31:16]															
	W																
	R	RXFIFO[15:0]															
	W																
0xBASE+0x004 (TXDATA)	R																
	W	TXFIFO[31:16]															
	R																
	W	TXFIFO[15:0]															
0xBASE+0x008 (CONTROLREG)	R	BURST LENGTH												CHANNEL SELECT		DRCTL	
	W																
	R	PRE DIVIDER				POST DIVIDER				CHANNEL MODE				SMC	XCH	HW	EN
	W																
0xBASE+0x00C (CONFIGREG)	R	0	0	0	HT LENGTH					SCLK_CTL				DATA_CTL			
	W																
	R	SSB POL				SSB CTL				SCLK POL				SCLK PHA			
	W																

Table 26-5. eCSPI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x010 (INTREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	TCEN	ROEN	RFEN	RDFEN	RREN	TFEN	TDREN	TEEN
	W																
0xBASE+0x014 (DMAREG)	R	RXTDEN	0	RX DMA LENGTH						RXDEN	0	RX WATER MARK					
	W																
	R	0	0	0	0	0	0	0	0	TXDEN	0	TX WATER MARK					
	W																
0xBASE+0x018 (STATUSREG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	TC	RO	RF	RH	RR	TF	TH	TE
	W									w1c	w1c						
0xBASE+0x01C (PERIODREG)	R	0	0	0	0	0	0	0	0	0	0	CSD CTRL					
	W																
	R	CSRC	SAMPLE PERIOD[14:0]														
	W																
0xBASE+0x020 (TESTREG)	R	LBC	0	CL		0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	RXCNT							0	TXCNT						
	W																
0xBASE+0x040 (MSGDATA)	R																
	W	MSGDATA[31:16]															
	R																
	W	MSGDATA[15:0]															

26.3.3 Register Descriptions

This section provides detailed descriptions of the eCSPI registers in address order.

26.3.3.1 RXFIFO DATA Register (RXDATA)

The RXFIFO Data register (RXDATA) forms the top word of 64 × 32 RXFIFO. Only word size operation is allowed for this register. Write to this register will be ignored, and read this register will get return value

from RXFIFO. Figure 26-3 shows the RXDATA Data register and Table 26-6 shows the register's field descriptions.

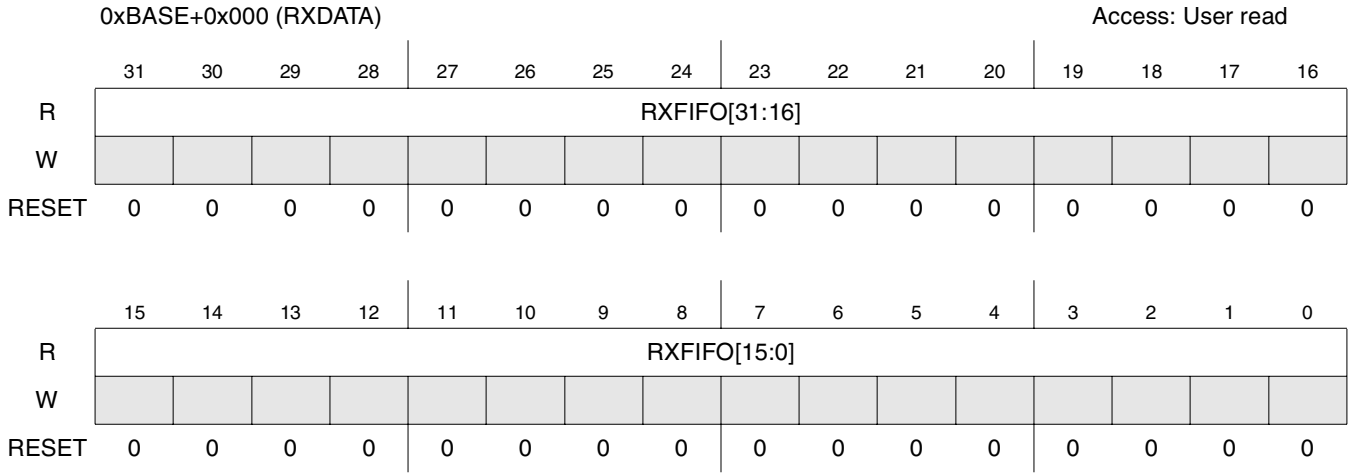


Figure 26-3. RXDATA Data Register Diagram

Table 26-6. RXDATA Data Register Field Description

Field	Description
31–0 RXDATA	RXDATA holds the top word of RXFIFO. The RXFIFO is advanced for each read of this register. The data read is zero when the Receive Data Ready (RR) bit in the Interrupt Control/Status register is cleared. The data write to this register will be ignored.

26.3.3.2 TXFIFO DATA Register (TXDATA)

The TXFIFO Data register (TXDATA) forms the top word of 64 × 32 TXFIFO. Only word size operation is allowed for this register. Read to this register will result zero, and write this register will store data into TXFIFO. Figure 26-4 shows the TXDATA Data register and Table 26-7 shows the register's field descriptions.

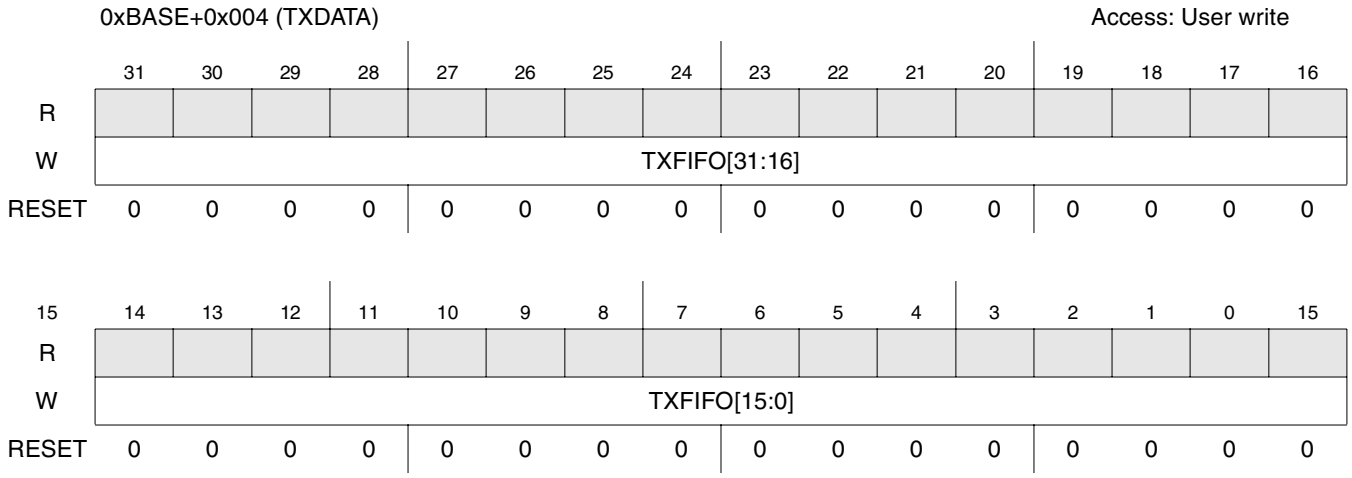


Figure 26-4. TXDATA Data Register Diagram

Table 26-7. TXDATA Data Register Field Description

Field	Description
31–0 TXDATA	TXDATA holds the top word of TXFIFO. The TXFIFO is advanced for each write of this register. The data read is zero. The data write to this register will be stored into TXFIFO.

26.3.3.3 Control Register (CONTROLEG)

The Control Register (CONTROLREG) allows the user to enable the eCSPI module, specify the clock divider value, select the SPI channel, configure $\overline{\text{SPI\_RDY}}$ control mode, and define the transfer length, etc. Figure 26-5 shows the Control register and Table 26-8 shows the register's field descriptions.

0xBASE+0x008 (CONTROLREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BURST LENGTH												CHANNEL SELECT		DRCTL	
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRE DIVIDER				POST DIVIDER				CHANNEL MODE				SMC	XCH	HW	EN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 26-5. eCSPI Control Register Diagram

Table 26-8. eCSPI Control Register Field Description

Field	Description
31–20 BURST LENGTH	Burst Length. This field defines number of bits to be transferred in one SPI burst. $\overline{SS}$ will remain asserted until all bits in a SPI burst are shifted out. A maximum of 2^12 bits can be transferred in a single SPI burst. In master mode, it controls the number of bits per SPI burst. Since the shift register always loads 32-bit data from TXFIFO, only the n least-significant (n = BURST LENGTH[4:0] + 1) bits in the first word will be shifted out. The remaining bits in first word will be ignored. All bits in other words will be shifted out. In slave mode, only when SSCTL is cleared, this field will take effect in SPI transfer. Number of Valid Bits in a SPI burst. 0x000 A SPI burst contains least 1 bit in a TXFIFO word. 0x001 A SPI burst contains least 2 bit in a TXFIFO word. 0x002 A SPI burst contains least 3 bit in a TXFIFO word. ..... 0x01F A SPI burst contains all 32 bits in a TXFIFO word. 0x020 A SPI burst contains least 1 bit in first TXFIFO word and all 32 bits in second TXFIFO word. 0x021 A SPI burst contains least 2 bits in first TXFIFO word and all 32 bits in second TXFIFO word. ..... 0xFFE A SPI burst contains least 31 bits in first TXFIFO word and 2^7 -1 TXFIFO words. 0xFFF A SPI burst contains 2^7 TXFIFO words.
19–18 CHANNEL SELECT	SPI CHANNEL SELECT bits. It selects which SPI channel is selected to be active. 00 SPI Channel 0 is selected. 01 SPI Channel 1 is selected. 10 SPI Channel 2 is selected. 11 SPI Channel 3 is selected.
17–16 DRCTL	SPI Data Ready Control. This 2-bit field selects the utilization of the $\overline{SPI\_RDY}$ in master mode. eCSPI will check this fields before it start a SPI burst. 00 Don't care $\overline{SPI\_RDY}$ 01 Burst will be triggered by falling edge of $\overline{SPI\_RDY}$. 10 Burst will be triggered by low level of $\overline{SPI\_RDY}$. 11 $\overline{RSV}$.
15–12 PRE DIVIDER	SPI Pre Divider bits. eCSPI uses a two stages divider structure to generated the SPI clock. This four-bit field defines the ipg_clk 4-bit pre-divider. 0000Divide by 1. 0001Divide by 2. 0010Divide by 3. ..... 1101Divide by 14. 1110Divide by 15. 1111Divide by 16.
11 –8 POST DIVIDER	SPI Post Divider bits. eCSPI uses a two stages divider structure to generated the SPI clock. This four-bit field defines the 4-bit post-divider. 0000Divide by 1. 0001Divide by 2. 0010Divide by 4. ..... 1110Divide by 2^14. 1111Divide by 2^15.

Table 26-8. eCSPI Control Register Field Description (continued)

Field	Description
7–4 CHANNEL MODE	SPI CHANNEL MODE defines mode of each SPI channel. CHANNEL MODE[0] defines the mode of SPI channel 0. CHANNEL MODE[1] defines the mode of SPI channel 1. CHANNEL MODE[2] defines the mode of SPI channel 2. CHANNEL MODE[3] defines the mode of SPI channel 3. 0 Slave Mode. 1 Master Mode.
3 SMC	Start Mode Control. This bit is used in master mode only and it controls how eCSPI start a SPI burst. 0 Normal mode, the XCH bit controls when a SPI burst can start. Write a 1 to XCH bit will start a SPI burst or multiple bursts. (controlled by SSCTL) 1 Automatic Mode, start a SPI burst when a data is written in TXFIFO immediately.
2 XCH	SPI Exchange Bit. If the SMC bit is cleared, writing a 1 to this bit starts one SPI bursts/multiple SPI bursts according to SSCTL bit. This bit remains set while either the exchange is in progress, or the eCSPI is waiting for active input if $\overline{\text{SPIRDY}}$ is enabled through DRCTL. This bit is cleared automatically when one SPI burst has been done (SSCTL is clear) or TXFIFO is empty (SSCTL is set). In Slave mode, this bit is ignored. 0 Idle. 1 Initiates exchange (write) or busy (read).
1 HW	HW Trigger Enable. This bit is used in master mode only and it enables hardware trigger mode. 0 HT Mode is disabled. 1 HT Mode is enabled.
0 EN	SPI Module Enable Control. This bit enables the eCSPI. This bit must be asserted before writing to other registers or initiating an exchange. Writing zero to this bit disables the module and resets the internal logic with the exception of the CONTROLREG. The module's internal clocks are gated off whenever the module is disabled. 1 eCSPI is enabled. 0 eCSPI is disabled.

26.3.3.4 Config Register (CONFIGREG)

The Config Register (CONFIGREG) allows the user to configure each channel of the eCSPI module, configure its operating modes, phase, and polarity of the clock, configure the $\overline{\text{SS}}$. The reserved bits are always read as zero.

Figure 26-6 shows the CONFIGREG register and Table 26-9 shows the register's field descriptions.

0xBASE+0x00C (CONFIGREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	HT LENGTH					SCLK CTL				DATA CTL			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSB POL				SSB CTL				SCLK POL				SCLK PHA			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 26-6. eCSPI Config Register Diagram

Table 26-9. eCSPI Config Register Field Description

Field	Description
31–29	Reserved, all bits can be read/write, but have no meaning.
28–24 HT LENGTH	The HT LENGTH defines the length of message content in HT Mode. The number of bits of one message content is HT LENGTH + 1;
23–20 SCLK CTL	The SCLK CTL control inactive state of SCLK line for each SPI channel. Bit 27 controls the SCLK inactive state of SPI channel 3. Bit 26 controls the SCLK inactive state of SPI channel 2. Bit 25 controls the SCLK inactive state of SPI channel 1. Bit 24 controls the SCLK inactive state of SPI channel 0. 0 stay high. 1 stay low.
19–16 DATA CTL	The Data CTL control inactive state of Data line for each SPI channel. Bit 23 controls the data polarity of SPI channel 3. Bit 22 controls the data polarity of SPI channel 2. Bit 21 controls the data polarity of SPI channel 1. Bit 20 controls the data polarity of SPI channel 0. 0 stay high. 1 stay low.
15–12 SSB POL	The SSB POL control polarity of SSB for each SPI channel. Bit 15 controls the SSB Polarity of SPI channel 3. Bit 14 controls the SSB Polarity of SPI channel 2. Bit 13 controls the SSB Polarity of SPI channel 1. Bit 12 controls the SSB Polarity of SPI channel 0. 0 Active Low. 1 Active High.

Table 26-9. eCSPI Config Register Field Description (continued)

Field	Description
11–8 SSB CTRL	The SSB CTL control behavior of SSB for each SPI channel. Bit 11 controls the SSB behavior of SPI channel 3. Bit 10 controls the SSB behavior of SPI channel 2. Bit 9 controls the SSB behavior of SPI channel 1. Bit 8 controls the SSB behavior of SPI channel 0. In Master mode, this bit defines if it is in multiple burst mode when SMC bit is cleared. When SMC bit is set, this bit will be ignored. 0 Single burst Mode. 1 Multiple bursts Mode. In Slave mode, this bit controls when a SPI burst is completed. 0 SPI burst completed when (BURST_LENGTH + 1) bits are received. 1 SPI burst completed when SSB input negated.
7–4 SCLK POL	The SCLK POL control polarity of SCLK for each SPI channel. Bit 7 controls the SCLK Polarity of SPI channel 3. Bit 6 controls the SCLK Polarity of SPI channel 2. Bit 5 controls the SCLK Polarity of SPI channel 1. Bit 4 controls the SCLK Polarity of SPI channel 0. 0 Active high polarity (0 = Idle). 1 Active low polarity (1 = Idle).
3–0 SCLK PHA	The SCLK PHA control data phase of SCLK for each SPI channel. Bit 3 controls the SSB Polarity of SPI channel 3. Bit 2 controls the SSB Polarity of SPI channel 2. Bit 1 controls the SSB Polarity of SPI channel 1. Bit 0 controls the SSB Polarity of SPI channel 0. 0 Phase 0 operation. 1 Phase 1 operation.

26.3.3.5 Interrupt Control Register (INTREG)

The Interrupt Control Register (INTREG) enables the generation of interrupts to the MCU. The reserved bits are always read as zero. Figure 26-7 shows the Interrupt Control register and Table 26-10 shows the register's field descriptions.

0xBASE+0x010 (INTREG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	TCEN	ROEN	RFEN	RDREN	RREN	TFEN	TDREN	TEEN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 26-7. Interrupt Control Register Diagram

Table 26-10. Interrupt Control Register Field Descriptions

Field	Description
31–8	Reserved, all bits should read zero.
7 TCEN	Transfer Completed Interrupt enable. This bit enables the Transfer Completed Interrupt. 0 Disable 1 Enable
6 ROEN	RXFIFO Overflow Interrupt enable. The bit enables the RXFIFO Overflow Interrupt. 0 Disable 1 Enable
5 RFEN	RXFIFO Full Interrupt enable. The bit enables the RXFIFO Full Interrupt. 0 Disable 1 Enable
4 RDREN	RXFIFO Data Request Interrupt enable. The bit enables the RXFIFO Data Request Interrupt when number of data in RXFIFO is great than RX WATER MARK when RXTDEN is clear. When RXTDEN is set, it will active as description in Table 26-11. 0 Disable 1 Enable
3 RREN	RXFIFO Ready Interrupt enable. The bit enables the RXFIFO Ready Interrupt. 0 Disable 1 Enable
2 TFEN	TXFIFO Full Interrupt enable. The bit enables the TXFIFO Full Interrupt. 0 Disable 1 Enable
1 TDREN	TXFIFO Data Request Interrupt enable. The bit enables the TXFIFO Data Request Interrupt when number of data in TXFIFO is no more than TX WATER MARK. 0 Disable 1 Enable
0 TEEN	TXFIFO Empty Interrupt enable. The bit enables the TXFIFO Empty Interrupt. 0 Disable 1 Enable

26.3.3.6 DMA Control Register (DMAREG)

The DMA Control Register (DMAREG) provides the user a way to access eCSPI with DMA. The eCSPI sends out DMA requests when the appropriate FIFO conditions are matched. The reserved bits are always read as zero.

Figure 26-8 shows the CNTRL register and Table 26-11 shows the register's field descriptions.

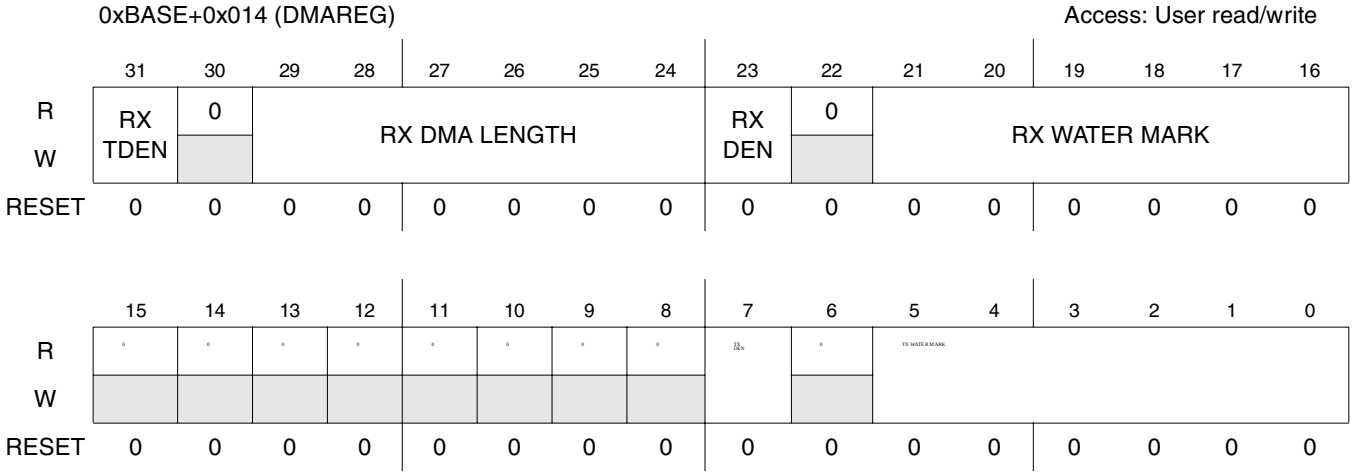


Figure 26-8. DMA Control Register Diagram

Table 26-11. DMA Control Register Field Description

Field	Description
31 RXTDEN	RXFIFO TAIL DMA Request Enable. When this bit is set, a internal counter is enabled and is increased at each read of RXFIFO. The counter will be cleared automatically when it reach RX DMA LENGTH. If number of words remained in RXFIFO is great or equal to RX DMA LENGTH - content of the counter, a DMA request will be generated even if it is less than RX WATER MARK. 0 Disable 1 Enable
30–27	Reserved, all bits should read zero.
26–24 RX DMA LENGTH	RX DMA LENGTH. These bits define how many words will be retrieved in one DMA access cycle. (Only used when RXTDEN is set.)
24 RXDEN	RXFIFO DMA Request Enable. This bit enables/disables the RXFIFO DMA Request. 0 Disable 1 Enable
23–22	Reserved, should be cleared.
21–16 RX WATER MARK	RX WATER MARK. These bits control when eCSPI issue a RX DMA request. A RX DMA/Interrupt request will be issued when number of data in RXFIFO is great than RX WATER MARK.
15–9	Reserved, should be cleared.
8 TXDEN	TXFIFO DMA Request Enable. This bit enables/disables the TXFIFO DMA Request. 0 Disable 1 Enable
7–6	Reserved, should be cleared.
5–0 TX WATER MARK	TX WATER MARK. These bits control when eCSPI issue a TX DMA request. A DMA/Interrupt request will be issued when number of data in TXFIFO is no more than TX WATER MARK.

26.3.3.7 Status Register (STATREG)

The eCSPI Status Register (STATUSREG) reflects the status of the eCSPI module operating condition. The reserved bits are always read as 0. If the eCSPI is disabled, this register reads 0x0000_0003. Figure 26-9 shows the Status register and Table 26-12 shows the register's field descriptions.

0xBASE+0x018 (STATUSREG) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	TC	RO	RF	RDR	RR	TF	TDR	TE
W									w1c	w1c						
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 26-9. Status Register Diagram

Table 26-12. Status Register Field Description

Field	Description
31–8	Reserved, should be cleared.
7 TC	Transfer Completed. When set, this bit indicates current transmission is completed. Writing 1 to this bit clears it. 0 Busy. 1 Transfer Completed.
6 RO	RXFIFO Overflow. When set, this bit indicates that RXFIFO has overflowed.Writing 1 to this bit clears it. 0 RXFIFO is available. 1 RXFIFO has overflowed.
5 RF	RXFIFO Full. This bit is set when the RXFIFO is full (64 words). 0 Not Full. 1 RXFIFO is Full. (64 words)
4 RDR	RXFIFO Data Request. When RXTDE is set, 0 Number of data in RXFIFO is less than RX DMA WATER MARK. 1 Number of data in RXFIFO is great than RX DMA WATER MARK or a DMA TAIL DMA condition is match. When RXTDE is clear, 0 Number of data in RXFIFO is less than RX DMA WATER MARK. 1 Number of data in RXFIFO is great than RX DMA WATER MARK.
3 RR	RXFIFO Ready. This bit is set any time there is one or more words stored in RXFIFO (≥ 1 words). 0 No valid data in RXFIFO. 1 At least 1 word in RXFIFO.
2 TF	TXFIFO Full. This bit is set when if the TXFIFO is full (64 words). 0 TXFIFO is not Full. 1 TXFIFO is Full.

Table 26-12. Status Register Field Description (continued)

Field	Description
1 TDR	TXFIFO Data Request. 0 Number of empty slots in TXFIFO is great than TX DMA WATER MARK. 1 Number of empty slots in TXFIFO is no more than TX DMA WATER MARK.
0 TE	TXFIFO Empty. This bit is set if the TXFIFO is empty. 0 TXFIFO contains one or more words. 1 TXFIFO is empty.

26.3.3.8 Sample Period Control Register (PERIODREG)

The Sample Period Control Register (PERIODREG) provides the user a way to insert delays (wait states) between consecutive SPI transfers. Control bits in this register select the clock source for the sample period counter and the delay count indicating the number of wait states to be inserted between data transfers. Delay counts are only applicable when the eCSPI module is operating in master mode. Also it contains CSD CTRL which is used to insert delay between Chip Select active edge and first SPI Clock edge.

Figure 26-10 shows the Sample Period Control register and Table 26-13 shows the register's field descriptions.

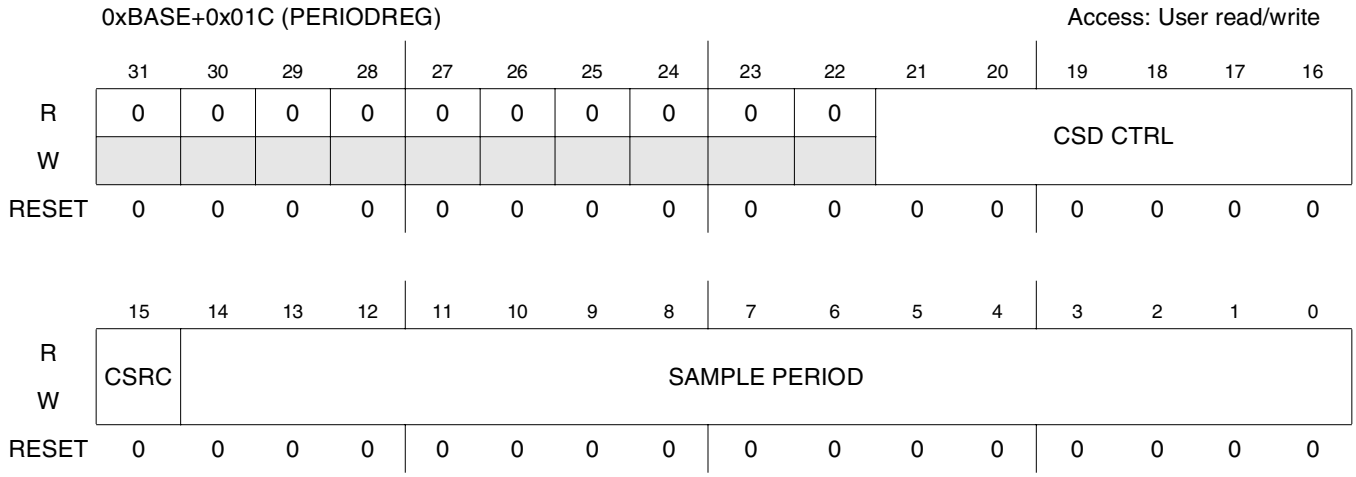


Figure 26-10. Sample Period Control Register Diagram

Table 26-13. Sample Period Control Register Field Description

Field	Description
31–22	Reserved, all bits should read zero.
21-16 CSD CTRL	Chip Select Delay Control bits. These bits define how many SPI clocks will be inserted between chip select active edge and first SPI clock edge.
15 CSRC	Clock Source Control. This bit selects the clock source for the sample period counter. 0 SPI Clock (SCLK) 1 CKIL (32.768 KHz)
14–0 SAMPLE PERIOD	Sample Period Control. These bits define the number of wait states to be inserted between data transfers.

26.3.3.9 Test Control Register (TESTREG)

The Test Control Register (TESTREG) provides the user a mechanism to internally connect the receive and transmit devices of the eCSPI module, display the status of the state machine, monitor the contents of the receive and transmit FIFO, and debug the eCSPI.

Figure 26-11 shows the Test Control register and Table 26-14 shows the register's field descriptions.

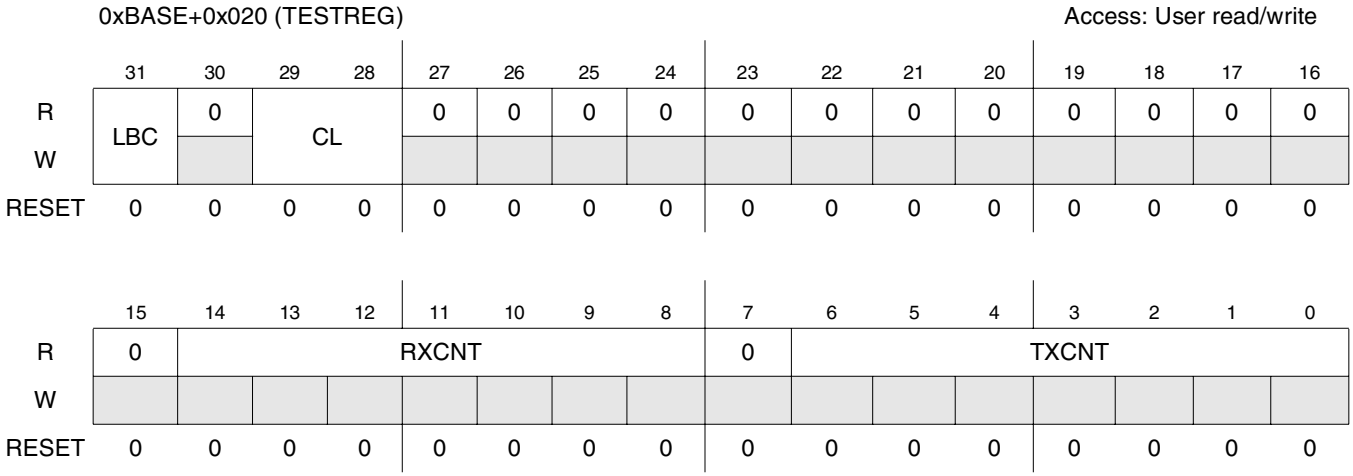


Figure 26-11. Test Control Register Diagram

Table 26-14. Test Control Register Field Description

Field	Description
31 LBC	Loop Back Control. This bit is used in Master mode only. When this bit is set, the eCSPI module connects the transmitter and receiver sections internally, and the data shifted out from the most-significant bit of the Shift register is looped back into the least-significant bit of the Shift register. In this way, a self-test of the complete transmit/receive path can be made. The output pins are not affected, and the input pins are ignored. 0 Not connected. 1 Internally connected.
30	Reserved, all bits should read zero.

Table 26-14. Test Control Register Field Description (continued)

Field	Description
29–28 CL	Catch Latency bits. These bits define the catch latency mode which will give more time on path delay between chip and external Slave devices. 00 Normal mode. 01 The data latch event will be delayed for half SPI clock cycle. 10 The data latch event will be delayed for one SPI clock cycle. 11 The data latch event will be delayed for one and half SPI clock cycle.
27–15	Reserved, all bits should read zero.
14–8 RXCNT	RXFIFO Counter. These bits indicate the number of words in RXFIFO.
7	Reserved, all bits should read zero.
6–0 TXCNT	TXFIFO Counter. These bits indicate the number of words in TXFIFO.

26.3.3.10 Message Data Register (MSGDATA)

The Message Data Register (MSGDATA) forms the top word of 16 × 32 MSG Data FIFO. Only word size operation is allowed for this register. Read to this register will result zero, and write this register will store data into MSGFIFO. Figure 26-12 shows the Message Data Register and Table 26-15 shows the register's field descriptions.The Message DATA FIFO contains message contents that are associated with HT input strobe signals. The content in each slot of MSG Data FIFO will be send out via eCSPI Channel 0 once a pulse edge of associated HT input signal is detected. First slot will be associated with HT input signal 0, the second will be associated with HT input signal1, etc.

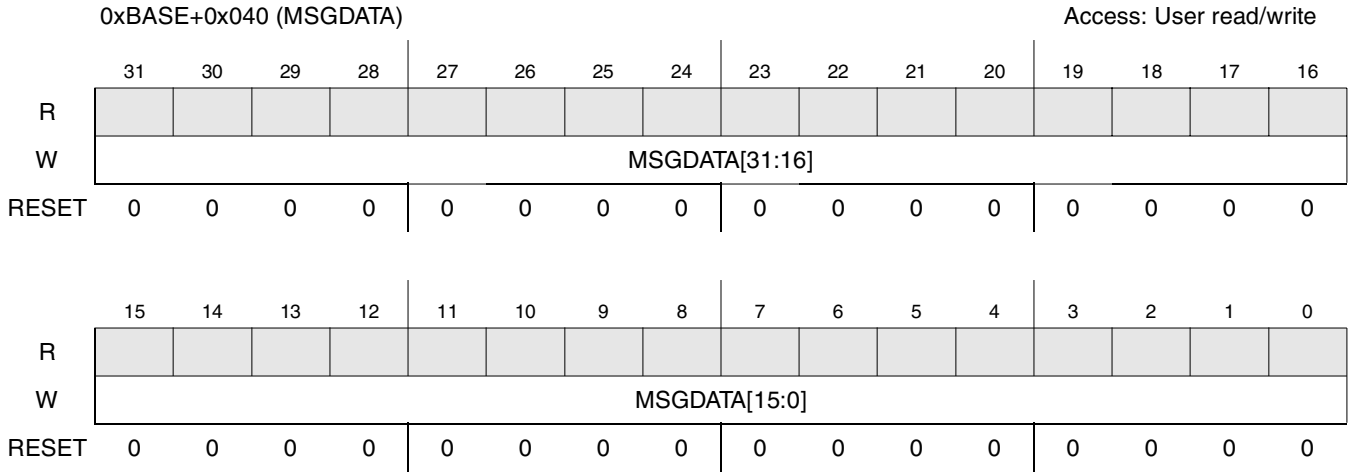


Figure 26-12. Message Data FIFO Diagram

Table 26-15. Message Data FIFO Field Description

Field	Description
31–0 MSGDATA	MSGDATA holds the top word of MSG Data FIFO. The MSG Data FIFO is advanced for each write of this register. The data read is zero. The data write to this register will be stored into MSG Data FIFO.

26.4 Functional Description

This section describes eCSPI module's functionality.

26.4.1 Clocks

There are total four clock sources in eCSPI, as follows:

- ipg_clk_s: used for IP bus read/write operations.
- ipg_clk: used for synchronize signals from ipg_clk_per domain.
- ipg_clk_per: used for baud clock generation and eCSPI operation clock.
- ipg_clk_32k: used for inserting wait states between SPI bursts when CSCR bit is set.

26.4.2 Reset

eCSPI can be reset by either a system (hard reset) or a software reset, as follows:

- For a system (hard) reset, reset eCSPI by asserting ipg_hard_neg_async_reset_b.
- For a software reset, reset eCSPI by clearing CSPI_EN bit in the control register. All registers except the control and config registers reset.

26.4.3 Interrupts

The following three kinds of conditions cause eCSPI to generate an interrupt:

- FIFO status—eCSPI will generate an interrupt which is associated with status of FIFOs. (Enable by set related bits in INT Register and DMA Register.)
- Transmit status—eCSPI will generate an interrupt when SPI bursts are finished.
- Error detected—eCSPI will generate an interrupt when RXFIFO is overflowed.

26.4.4 Endianness

eCSPI only supports little-endian mode.

26.4.5 eCSPI Timing Diagram

Figure 26-13 and Figure 26-14 show the generic eCSPI timing.

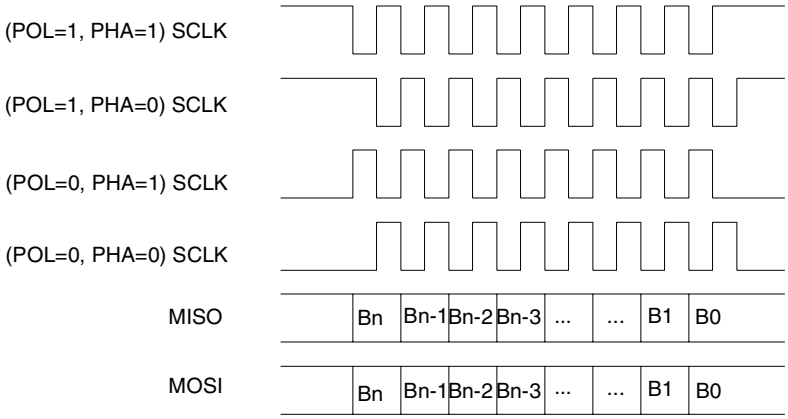


Figure 26-13. eCSPI Generic Timing

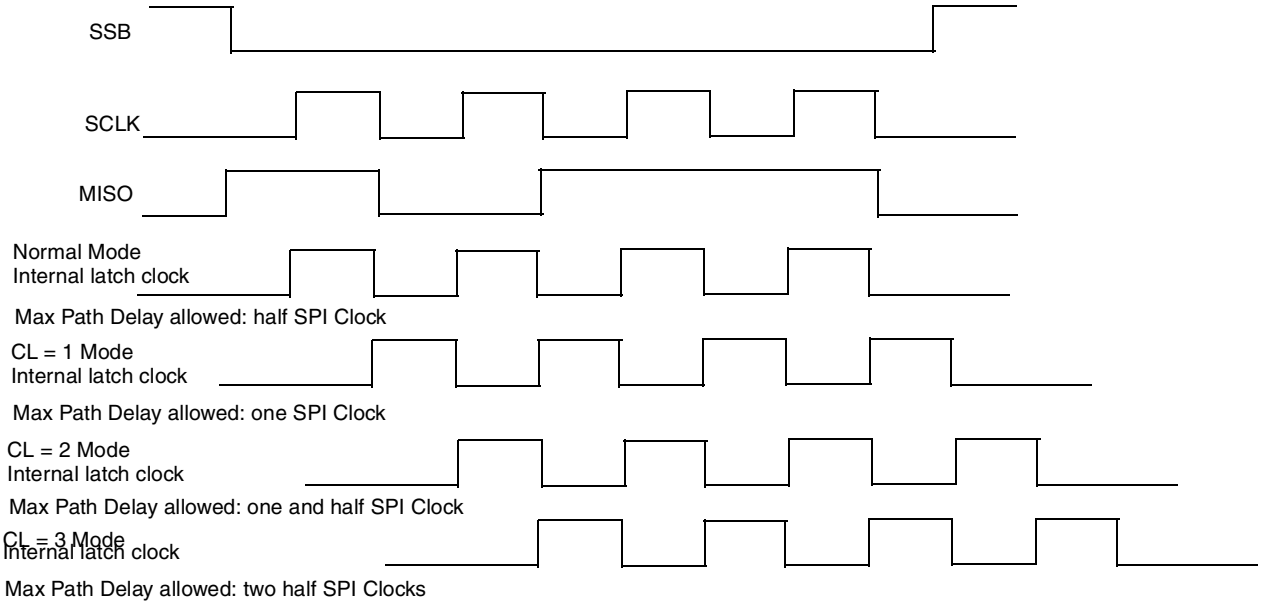


Figure 26-14. eCSPI Catch Latency Control Timing Diagram

26.4.6 Master Mode

The eCSPI master uses the $\overline{SS}$ signal to enable an external SPI device and uses SPICLK to transfer data in and out of the Shift register. The $\overline{SPI\_RDY}$ enables fast data communication with fewer software interrupts. By using PERIODREG, the eCSPI can be used for a fixed data transfer rate.

A SPI burst is defined as a bus transaction that starts when the slave select is asserted and ends when the slave select is deasserted. The SPI clock may not run continuously during the burst depends on status of TXFIFO.

When eCSPI is in Master mode the $\overline{SS}$, SCLK, and MOSI are output signals and the MISO is an input.

Figure 26-15 shows a typical SPI burst.

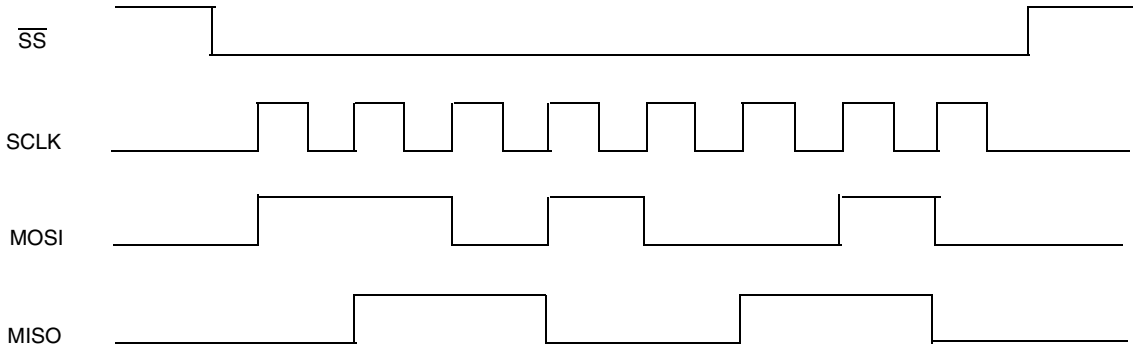


Figure 26-15. Typical SPI Burst (8-bit transfer)

In Figure 26-15, the $\overline{SS}$ signal enables the selected external SPI device and the SCLK synchronizes data transfer. MOSI and MISO are changed on rising edge of SCLK and the MISO is latched on the falling edge of the SLCK clock. The data shifted out is 0xD2, and the data shifted in is 0x66.

26.4.6.1 Master Mode with $\overline{SPI\_RDY}$

A SPI burst begins in master mode when following conditions have been satisfied:

1. When CONTROLREG[DRCTL] is 00 or 11, eCSPI is enabled, TXFIFO contains data, then CONTROLREG[XCH] or CONTROLREG[SMC] is set.
2. When CONTROLREG[DRCTL] is set to 01, a SPI burst will be triggered when a falling edge of $\overline{SPI\_RDY}$ has been detected. (Also condition 1 must be satisfied). Figure 26-16 shows the relationship between a SPI burst and the falling edge of $\overline{SPI\_RDY}$.

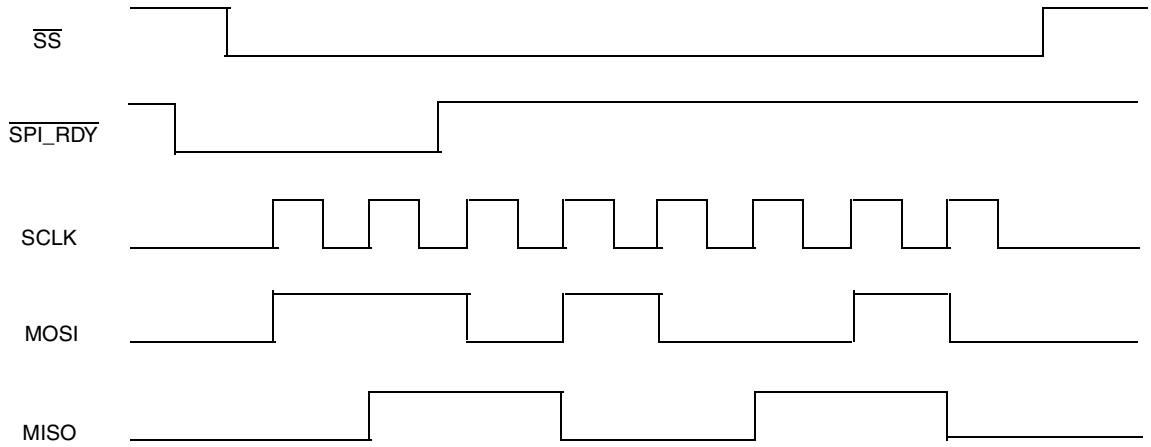


Figure 26-16. Relationship between a SPI Burst and the Falling Edge of SPI_RDY

The next SPI burst does not start until another falling edge of SPI_RDY is detected after the last burst has finished.

3. When CONTROLREG[DRCTL] is set to 10, a SPI burst will be triggered when SPI_RDY is low.(Also condition 1 must be satisfied). Figure 26-17 shows the relationship between a SPI burst and SPI_RDY. The SPI burst does not begin until SPI_RDY goes low. The next SPI burst begins after the last burst has finished if SPI_RDY remains low.

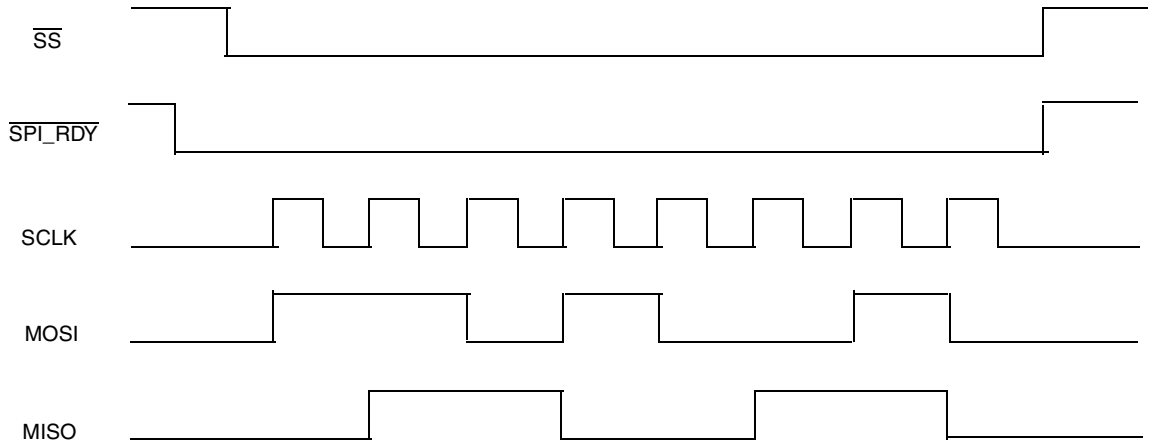


Figure 26-17. Relationship between a SPI Burst and SPI_RDY

26.4.6.2 Master Mode with Wait States

Wait states can be inserted between SPI bursts. This provides a way for the user to slow down the SPI burst to meet the timing requirements of a slower SPI device. Figure 26-18 shows wait states inserted between SPI bursts.

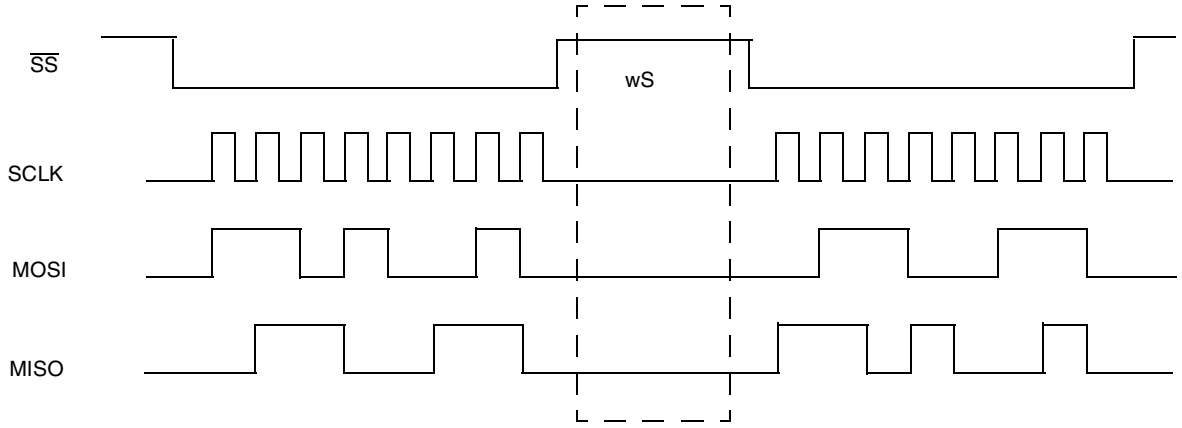


Figure 26-18. SPI Bursts with Wait States

In this case, the number of wait states is controlled by PERIODREG[SAMPLE PERIOD] and the wait states' clock source is selected by PERIODREG[CSRC].

26.4.6.3 Master Mode with CSD

Wait states can be inserted between SSB active and first SPI clock edge. This provides a way for the user to slow down the SPI burst to meet the timing requirements of a slower SPI device. Figure 26-19 shows the detail timing.

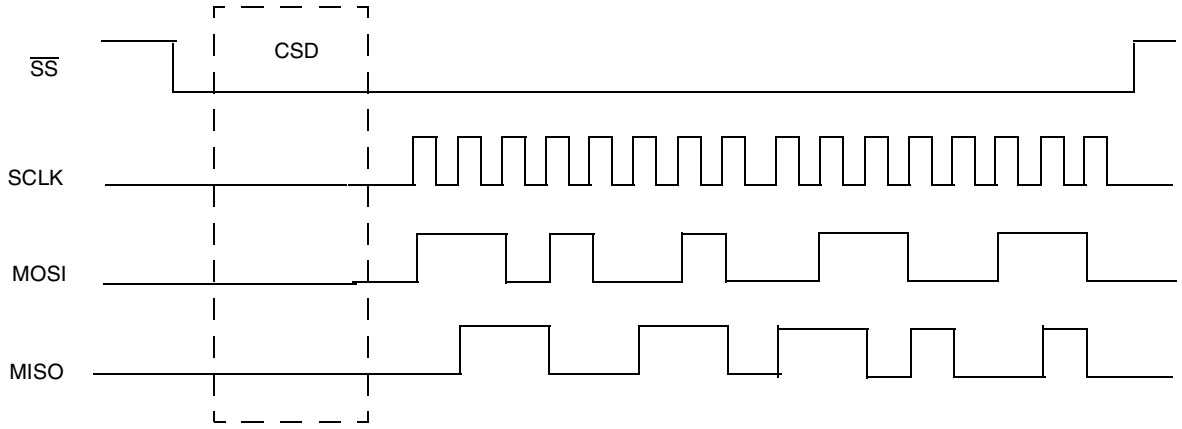


Figure 26-19. SPI Bursts with CSD

In this case, the number of wait states is controlled by PERIODREG[CSD] and the wait states are countered by SPI Clock.

26.4.6.4 Master Mode with SSCTL Control

SSCTL controls whether current operation is a single burst or multiple bursts in master mode. This bit will be ignored when SMC bit is also set. When SSCTL is set, current operation is multiple bursts transfer. When SSCTL is cleared, current operation is single burst transfer. The length of a SPI burst is defined in BURST LENGTH.

Figure 26-20 shows one SPI burst while SSCTL is clear.

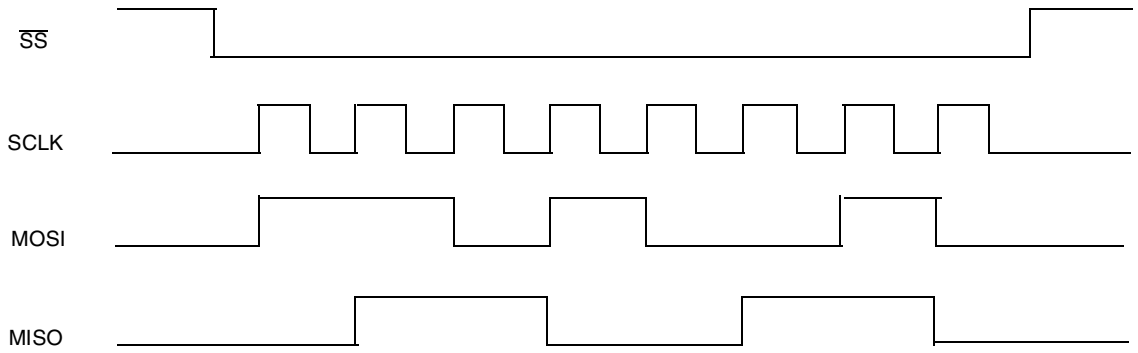


Figure 26-20. SPI Burst while SSCTL is Clear (One Burst)

In this case, two words are combined in one SPI burst and have been transmitted when CONTROLREG[XCH] is set. And CONTROLREG[XCH] is cleared after this SPI burst is completed.

Figure 26-21 shows two SPI bursts are transmitted while SSCTL is set.

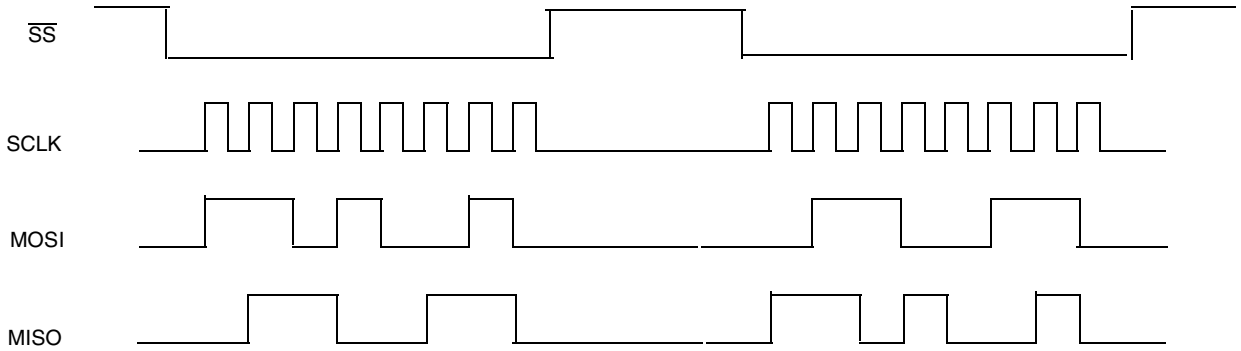


Figure 26-21. SPI Bursts while SSCTL is Set (Multiple Burst)

In this case, two SPI Bursts have been transmitted. When CONTROLREG[XCH] is set, eCSPI starts to transmit data between external device. Since the SSCTL is set, eCSPI will continue the transmission until the TXFIFO is empty. The CONTROLREG[XCH] will also keep set until TXFIFO is empty. If the wait states are inserted between SPI bursts, the SS will negate until the wait states finished. If there is no wait states inserted, the minimum width of SS pulse is 3 SPI clocks.

26.4.6.5 Master Mode with PHA Control

CONTROLREG[PHA] controls how the transmit data shifts out and the receive data shifts in.

When CONTROLREG[PHA] is set, the transmit data will shift out on the rising edge of SCLK, and the receive data is latched on the falling edge of SCLK. The most-significant bit is output on the first rising SPICLK edge.

When CONTROLREG[PHA] is cleared, the transmit data is shifted out on the falling edge of SCLK and the receive data is latched on the rising edge of SCLK. The most-significant bit is output when the CPU loads the transmitted data.

Inverting the SPICLK polarity does not impact the edge-triggered operations because they are internal to the serial peripheral interface master. Figure 26-22 shows a SPI burst using different POL and PHA configurations.

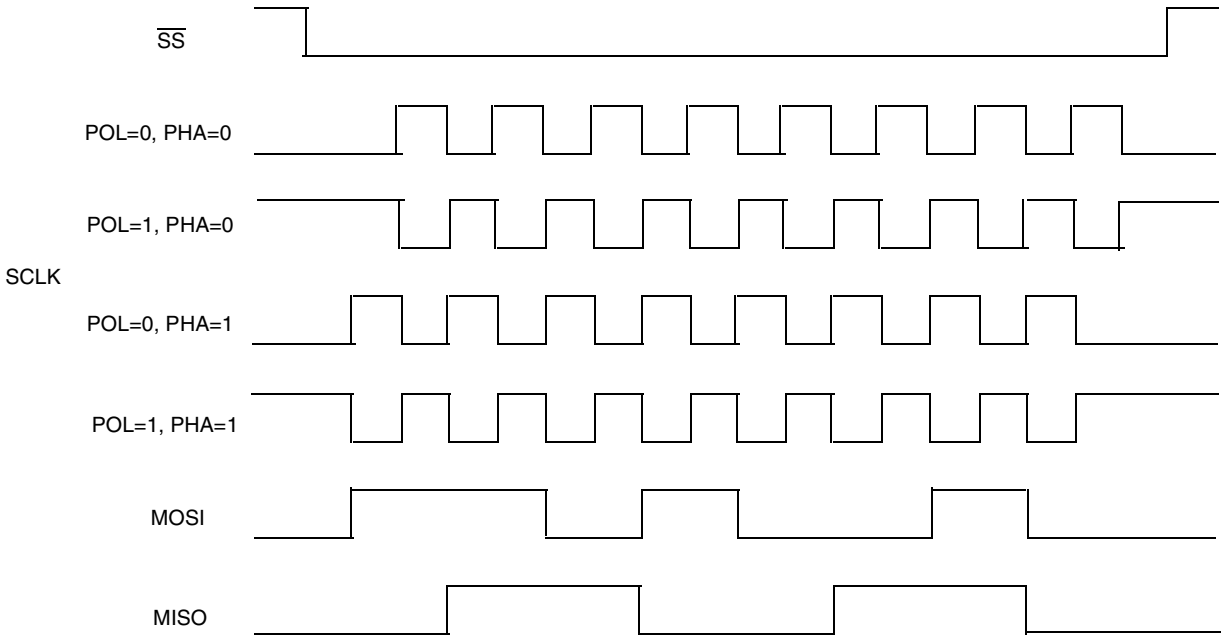


Figure 26-22. SPI Burst with Different POL and PHA Configuration.

26.4.7 Slave Mode

When the eCSPI module is configured as a slave, the user can configure the eCSPI Control register to match the external SPI master's timing. And the frequency of SPI clock input must no higher than quarter of ipg_clk_per's. In this configuration, $\overline{SS}$ becomes an input signal, and is used to latch data into or load data out to the internal data Shift registers, as well as to increment the data FIFO.

The $\overline{SS}$, SCLK, and MOSI are inputs and MISO is output. Most of their timing diagrams are the same as in Master mode, because the inputs come from a SPI master device.

However, it is different when $\overline{SS}$ is used to increment data FIFO. When the SSCTL is set while eCSPI is in Slave mode, the data FIFO will increment at $\overline{SS}$ rising edge (when SSPOL = 1, should be falling edge).

Figure 26-23 shows a SPI burst in which data FIFO is advanced by $\overline{SS}$ rising edge.

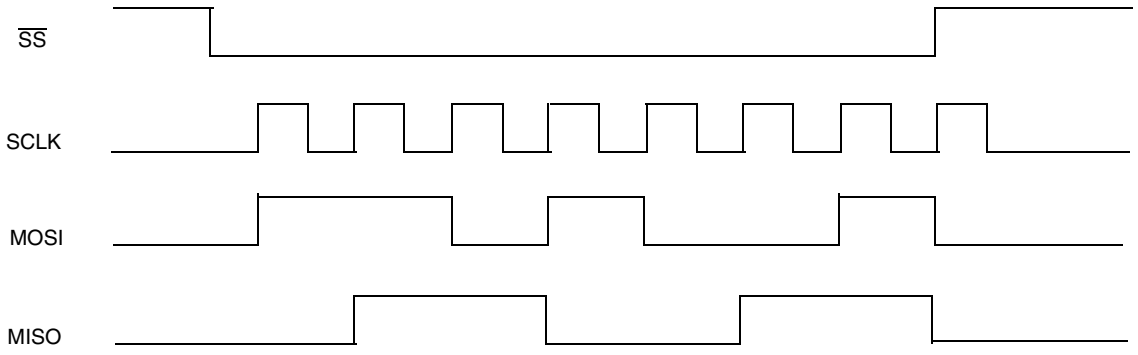


Figure 26-23. Increment Data FIFO by $\overline{SS}$ Rising Edge

In this case, the data received is not 0xD2 but 0x69. Only the most significant 7 bits are loaded to RXFIFO.

26.4.8 Hardware Trigger Mode

When CONTROLREG[HT] is set, eCSPI is enabled with HT Mode. This mode is similar with other master modes. In this mode, the burst length is limited in 32-bit. (Configurable from 1-bit to 32-bit). Also only SPI channel 0 can be used in HT mode. The main difference between HT Mode and other master modes is it's trigger mode. A SPI burst in HT Mode is triggered by HT input strobe signal other than a write to eCSPI register. So the SPI burst can be started with minimal CPU intervention for some timing critical case. In this mode, eCSPI must be configured as master mode. User must store message contents before enable this mode. Also only write operation is supported. Each message content associated with one hardware trigger input signal. Hardware trigger input signal 0 has the highest priority and hardware trigger input signal 15 has lowest priority. For example, in a write operation (16-bit command/data), the Burst Length bit in COFIGREG[HT LENGTH] should be set to 15, and effect data should be stored at least-significant 16-bit of Message Content n Register. CONTROLREG[HT] is set. After a hardware trigger has been detected, a 16-bit SPI Burst will be sent out to external device.

26.5 Initialization Information

This section provides initialization and application information for eCSPI.

Figure 26-24 shows two flow charts for the master and slave mode of operations supported by the eCSPI.

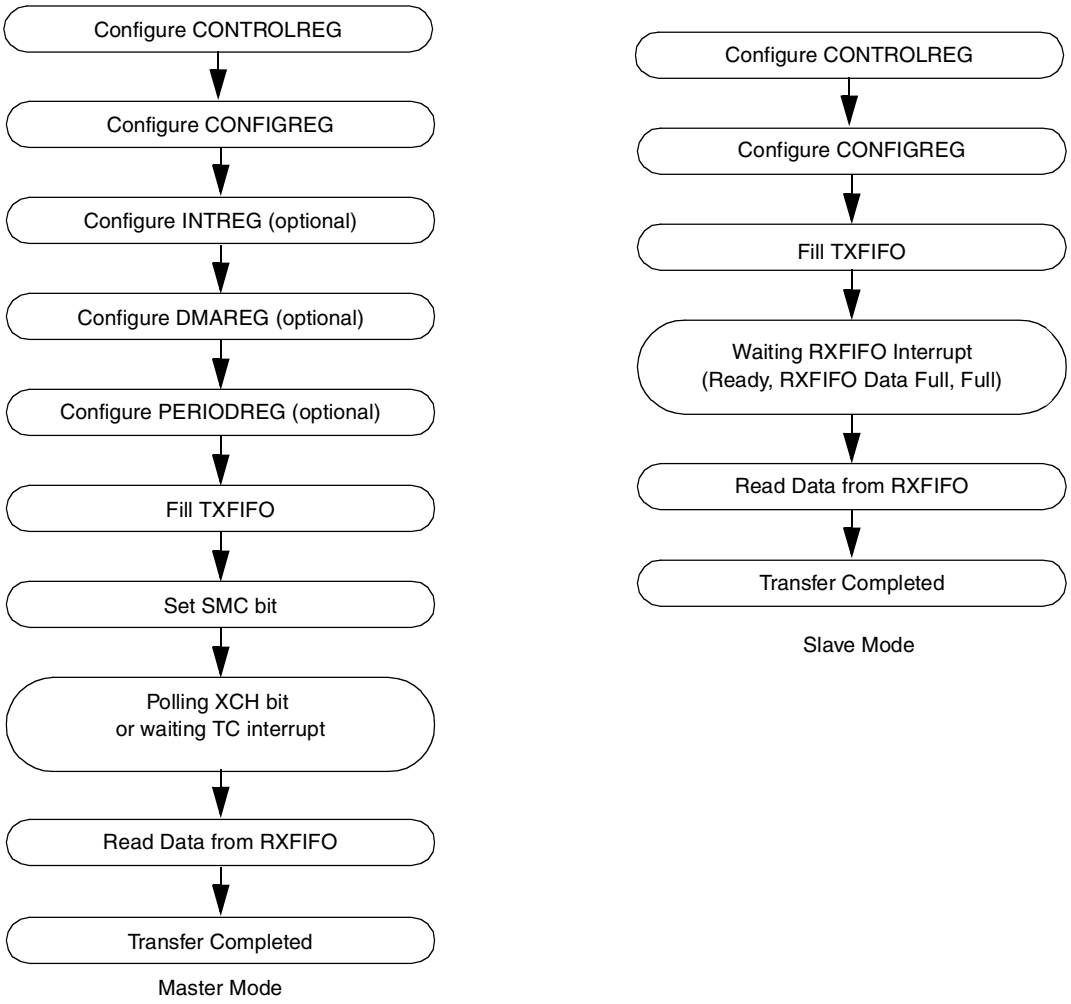


Figure 26-24. Flow Chart of eCSPI Operation

Example 26-1 shows a normal example code of eCSPI operation using C instructions.

Example 26-1. eCSPI Operation using C Instructions

```
//32bit burst, channel 0 is in master mode
mem32_write (eCSPI_CONTROL, 0x01F00011);
//Configure channel 0 to default configuration
mem32_write (eCSPI_CONFIG, 0x00000000);
//Enable interrupt if needed.
mem32_write (eCSPI_INT, 0x00000000);
//Enable DMA if needed
mem32_write (eCSPI_DMA, 0x00000000);
//Enable inserting wait states if needed
mem32_write (eCSPI_PERIOD, 0x00000000);
//Fill data into TXFIFO by instruction
for (i = 0; i <= 7; i = i + 1) {
 mem32_write (eCSPI_TXFIFO, 0x12345678 + i * 0x11223344);
}
```

```
//Set SMC bit to start transfer
mem32_write (eCSPI_CONTROL, 0x01F00019);
//Polling TC bit
while ((mem32_read (eCSPI_STATUS) & 0x80) == 0);
for (i = 0; i <= 7, i = i + 1) {
 mem32_read (eCSPI_RXFIFO);
.....                                  //Compare with expected value.
}
```

26.6 Application Information

This section discusses the following: interrupt control and DMA control.

26.6.1 Interrupt Control

Interrupt control is not a specific mode of operation; however, it provides a basic method to utilize the eCSPI FIFOs.

The eCSPI can be programmed to enable the TXFIFO empty, TXFIFO Data Request, and TXFIFO full interrupt. The interrupt service routine can also be used to fill the TXFIFO with data to be transferred. Furthermore, RXFIFO ready, RXFIFO Data Request, and RXFIFO full can be enabled to retrieve data from RXFIFO by using the interrupt service routine.

Three other interrupt sources can be used to control/debug the SPI bursts. The transfer-completed interrupt tells the user that there is no data left in TXFIFO and the data in the Shift register is shifted out. The RXFIFO overflow interrupt tells the user that the RXFIFO received more than 64 words and will not accept any other word.

Figure 26-25 shows a program sequence of SPI bursts using interrupt.

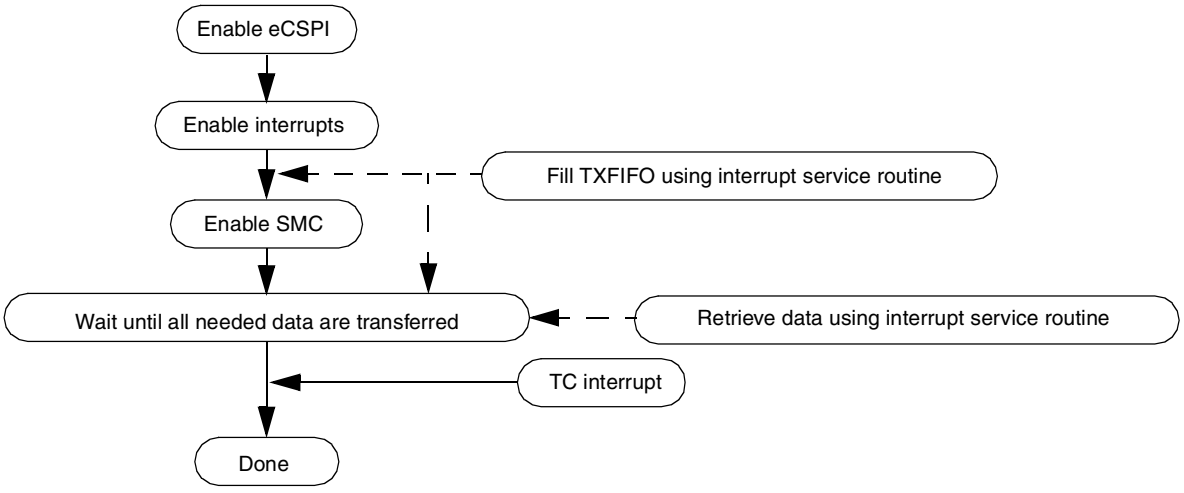


Figure 26-25. Program Sequence of SPI Burst Using Interrupt

26.6.2 DMA Control

DMA control provides another way to utilize the FIFOs in the eCSPI module. Peripherals that support DMA, such as the eCSPI, use DMA request and acknowledge signals. Larger amounts of data can be transferred using DMA control, thereby reducing interrupts and CPU loading. When the appropriate conditions are matched, the module sends out a DMA request, and the DMA deals with the following cases: TXFIFO empty, TXFIFO Data Request, RXFIFO Data Request, and RXFIFO full.

Figure 26-26 shows a program sequence of SPI bursts using the DMA.

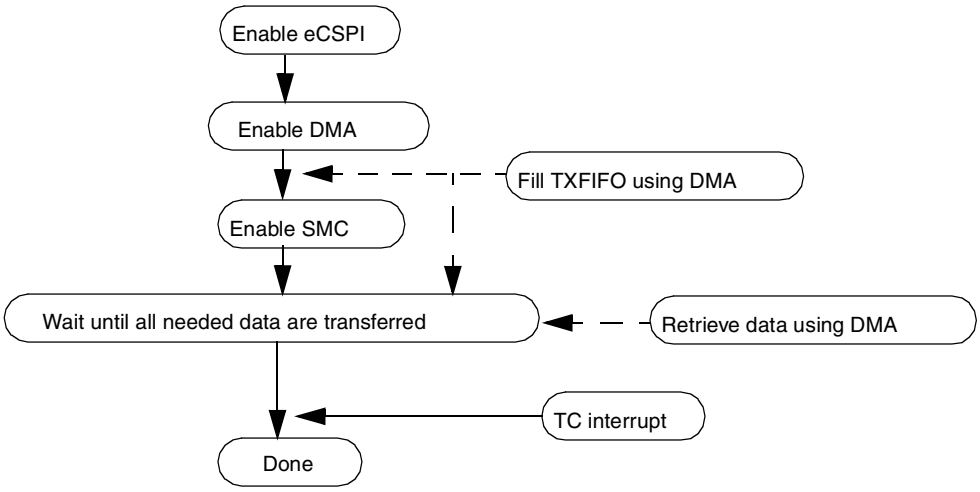


Figure 26-26. Program Sequence of SPI Burst Using DMA

Chapter 27
External Memory Interface (EMI)

27.1 Introduction

The EMI is a memory controller of memory devices in the system, both internal and external. It provides a capability for the system to control the external memory device, by preforming several type of accesses like read, write, program, and erase as well as internal memories of the system. All access from the system to a memory device are arbitrated inside EMI by M4IF submodule to the correct memory controller to perform the access. The high-level block diagram is presented in Figure 27-1.

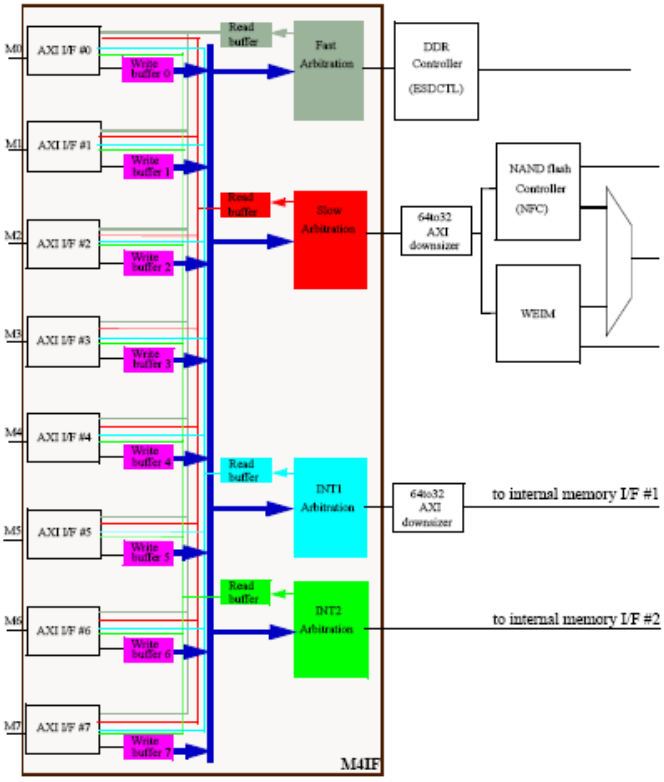


Figure 27-1. EMI Block Diagram

27.1.1 Overview

The EMI is an external memory interface with arbitration logic between multiple AXI masters to multiple memory controllers. It is divided into four major channels: a fast memories channel (Mobile SDR/DDR

SDRAM), a slow memories channel (NOR-FLASH/PSRAM/NAND-FLASH, and others.), and two internal memory channels (RAM, ROM).

To increase performance, the EMI separates the buffering and arbitration between accesses to fast channel slow channel and internal memory channels. Therefore, parallel accesses can occur. By separating these three channels, slow accesses do not interfere with fast accesses. EMI has been designed so that accesses by a slow master with a narrow bus do not block the access by a fast master with a wide bus.

27.1.2 Features

The EMI module includes the following features:

- Multi Master Multi Memory Interface (M4IF)
  - Supports multiple accesses from 8 masters through different input ports interfaces. Each port can support either of the following two data width options:
    - ×32 AXI port.
    - ×64 AXI port.
  - Supports different clock domains for each AXI port master.
  - Configurable memory snooping.
  - Configurable memory watermark protection per CS
  - Enhanced arbitration scheme for fast channel, consider page hit/miss, last access details(read/write), and fixed priority configuration.
- Enhanced SDRAM Controller (ESDRAMC) or LPDDR Controller (LPDDRC)
  - Up to 2 chip selects supporting up to 256 MByte each.
  - ×64 AXI port.
  - Support for ×16/×32 SDR SDRAM at clock frequencies up to 133 MHz.
  - Support for ×16/×32 LPDDR/Non-mobile DDR1 SDRAM at clock frequencies up to 166 MHz (DDR333).
  - Supports latency hiding logic.
- NANDFlash Controller—(NFC)
  - Supported pages sizes are ½ Kbyte, 2 Kbytes, and 4 Kbytes
  - Pages per blocks supported are 32, 64, 128, and 256
  - Up to 6 address cycles
  - 4.5K RAM Internal Buffer
  - MLC and SLC memory support.
  - Configurable operation mode - symmetric and asymmetric.
  - Configurable page mode, ½ Kbyte, 2 Kbytes, and 4 Kbytes
  - ECC support up to 8 bit.
  - Support of up to 8 mutually exclusive, yet interleaved..
- Wireless External Interface Memory Controller—(WEIM)
  - Up to 6 chip selects.

- Support ×32/×16 PSRAM (up to 133 MHz).
- support ×32/×16 muxed mode PSRAM/NOR (up to 133 MHz).
- Support ×32/×16 NOR (up to 133 MHz).

- Support DVFS, voltage, and frequency change.
- Support watermark configuration.
- Enhanced debug capability

27.1.3 Modes of Operation

The following sub sections describes several operation modes of EMI. For more information about the operation modes of each submodule in EMI, please refer to the “Modes of Operation” section in each submodule chapter.

27.1.3.1 Low-Power Modes

SoC system control provides an LPM request signal to EMI in order to let EMI enter into low-power mode. At the end of low-power mode sequence, EMI sends back an acknowledge signal to inform the system it entered into low-power mode.

In this mode, all internal clocks in EMI and its submodules are disabled by the system clock controller. Low-power mode definition is described in more details in each submodule separately.

27.1.3.2 Debug Mode

Several internal signals are routed out by the EMI debug unit to give the SoC the capability to track the internal logic mainly for the arbitration mechanism and memory served accesses. The debug unit and its debug signals are described in the M4IF chapter.

27.1.3.3 DVFS Support

EMI provides a real-time support of voltage and/or frequency change to its memory controllers. There is a handshake mechanism between EMI and the system so that the EMI is informed of any frequency change that is to be made in the system. When this handshake signal is sent to EMI, it stops all access to the external/internal memories controllers (READY signal of the memory controller will be set to LOW) and finishes any served access in the memory controller.

After that, the frequency change routine inside EMI is active so that at the end of the routine an acknowledge would be sent to inform a change clock frequency is allowed. At the moment the frequency was changed, the system should lower the handshake signal to EMI which means the frequency change is done. EMI would force a new measure in the delay line and move back to idle after the measure unit is updated with the new value (ready signals of the memory controlers will return to HIGH).

Please refer to each EMI submodule for detailed description of DVFS support.

Voltage change should be supported on the fly based on trigger signal from the system that inform EMI on a voltage change in the system. As a result, EMI would higher the measurement rate to allow better tracking of the voltage change and to allow system functionality in parallel to voltage change. When the

voltage change is done, the system lowers the trigger to EMI so that the measurement rate can be changed back to its default mode.

More details on delay line and measurement unit can be found in the “ESDCTL Spec” chapter.

27.1.3.4 Power Saving Mode

The EMI design supports few power saving modes other than LPMD. The M4IF module provides information about whether there are or are not pending requests on a certain arbitration when was the last request in this arbitration etc. Based on that info, both the specific arbitration and its correspond memory controller, would be able to enter power saving modes like Self refresh and Auto Self refresh in ESDCTL memory controller.
In this mode, each arbitration path is considered as a separate domain so that in a certain time internal memory path can be in power saving mode while DDR memory path will continue to work. It is important to mention that each power saving mode would leave an active logic to be able to get out of this mode whenever a new access arrives. In this case it would take few more cycles to serve the request since getting out of power saving mode routine is taking place first.

A quiet similar to that mechanism there is a similar mechanism on each of M4IF gaskets. M4IF can automatically shut off the clock of a specific gasket based on internal timer that counts a certain number of idle cycle.

Please refer to M4IF Chapter for more details on Power Saving mode.

27.1.3.5 SRPG Mode

EMI support SR Power gating mode. In this mode even when the power is gated off, several registers configuration in the memory controllers module will be kept so that when getting out of this mode the controller will be configured with the same values as it was before gating off the power. The following section list the register names that will be of SRPG type.

27.1.3.5.1 SRPG Register List

The following registers in ESDCTL module will be of SRPG type:

- ESDCTL0
- ESDCTG0
- ESDMISC

This configuration will give the ability to keep CS0 configuration as it was before gating off the power. Other CS will be disabled and will return to reset state.

The following registers in WEIM module will be of SRPG type:

- CS0GCR1
- CS0GCR2
- CS0RCR1
- CS0RCR2
- CS0WCR

- WIAR
- CS0WCR2

This configuration will give the ability to keep CS0 configuration as it was before gating off the power. Other CS will be disabled and will return to reset state.

Entering into SRPG mode should go through LPMD mode first, this is essential to the memory controller for right operation.

There are several registers in M4If which are SRPG type as well, mainly to support correct functionality after power gating.

27.2 Sharing of I/O Pins

The EMI module does not contain any I/O pins sharing or muxing between different arbitration paths. The only I/O sharing is done inside each arbitration/memory controller path, such as DDR and SDR I/O pin sharing or WEIM and NFC I/O pin sharing.

DDR and SDR I/O pin sharing is described in the ESDCTL chapter. WEIM and NFC I/O pin sharing is described below.

27.2.1 WEIM and NFC I/O Pin Sharing

The EMI contains an IOMUX submodule to be able to share 16-bits DATA pins of WEIM and NFC (slow arbitration). In order to be able to share data bus between these two memory controllers, data_dir signals areas are also muxed. The IOMUX module muxs between data BGA contacts of WEIM and NFC on the output path only, based on which memory controller owns the bus for the specific transaction.

WEIM's CRE signal is muxed inside the WEIM with one of the 24th–27th bits of the WEIM address, but only with the one that must be the highest available bit of the address and that is defined by the value of 4 vias bit_24_cre_via...bit_27_cre_via). For additional details on the WEIM_CRE signal, please refer to the WEIM Chapter.

Figure 27-2 shows data BGA contact sharing functionality between WEIM and NFC.

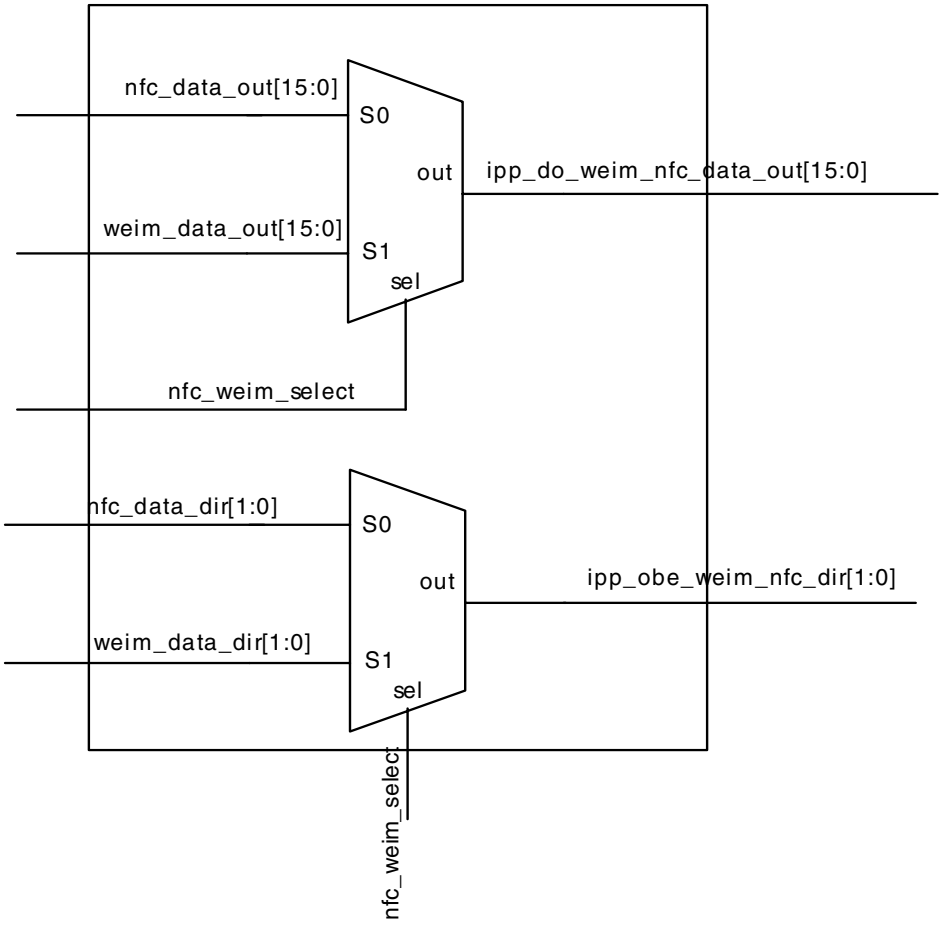


Figure 27-2. EMI IOMUX

The sharing control logic is based on the fix priority arbitrating between WEIM and NFC. WEIM has the priority on the external bus whenever there is a conflict between WEIM and NFC. However, when NFC is using the bus and WEIM access is issued in parallel, NFC frees up the bus after number of clock cycles depend on the specific command/access type is currently served in NFC. No specific time period is guaranteed for NFC to free up the bus for WEIM pending access.

27.2.2 WEIM/NFC Arbitration Logic

As was mentioned above there is a fixed priority between WEIM and NFC. The arbitration between these modules is done by a request and acknowledge signals from both modules.

NFC has two signals, a request signal from WEIM and an acknowledge signal to WEIM. Whenever NFC uses the bus it leaves the ack signal low until it finishes the transaction. If NFC does not use the bus, its ack signal will be high.

On the other side, WEIM has two signals as well, weim_grant and weim_busy. Whenever WEIM issues an access on the external bus, it needs to get weim_grant signal as high to enable its priority on the bus. If weim_grant is high, WEIM can issue the access and it is guaranteed NFC is not using the bus. Whenever WEIM gets an AXI access request it must assert weim_busy. weim-busy must remain asserted as long as the transaction isn't completed (also during the time the WEIM is waiting for weim_grant to be asserted).

Figure 27-3 is an high level scheme of the sub modules connectivity in EMI.

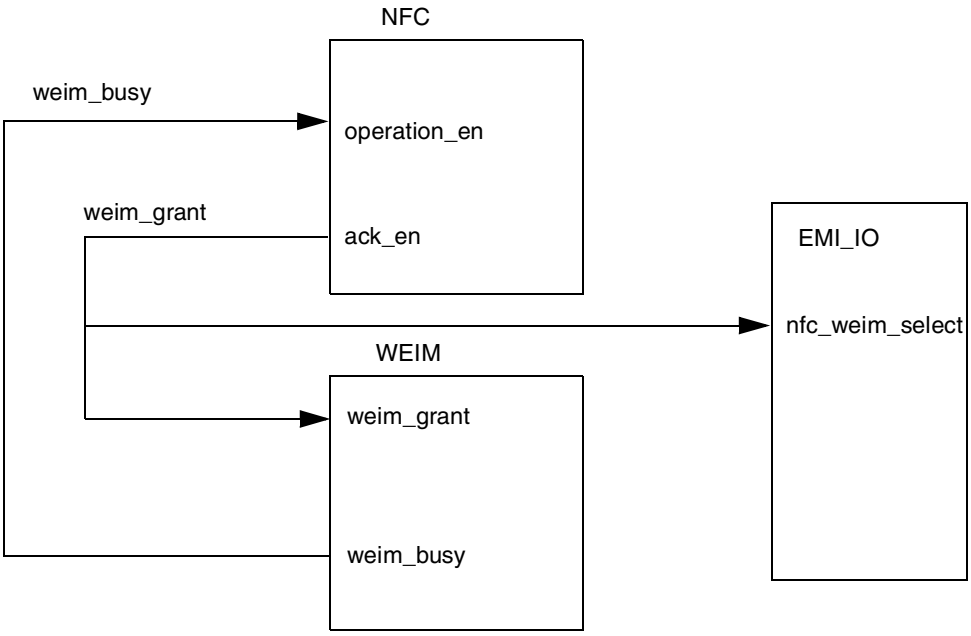


Figure 27-3. WEIM/NFC Arbitration Signals

27.3 External Signal Description

This section describes the signals that do, or may, connect off chip, and use a BGA contact. Table 27-2 describes the signal list of EMI off chip interface.

Table 27-1. EMI Signal Properties

Pin	Description	I/O
aclk_fast	Input clock-fast arbit	Input
aclk_int2	Input clock-inter2 mem.	Input
aclk_intr	Input clock-inter mem.	Input
aclk_m0	Input clock - master	Input
aclk_m1	Input clock - master	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
aclk_m2	Input clock - master	Input
aclk_m3	Input clock - master	Input
aclk_m4	Input clock - master	Input
aclk_m5	Input clock - master	Input
aclk_m6	Input clock - master	Input
aclk_m7	Input clock - master	Input
aclk_slow	Input clock-slow arbit	Input
act_cs[5:0]	weim via	Input
addrs0[1:0]	weim via	Input
addrs1[1:0]	weim via	Input
addrs2[1:0]	weim via	Input
addrs3[1:0]	weim via	Input
addrs4[1:0]	weim via	Input
addrs5[1:0]	weim via	Input
ap_via[4:0]	EMI IPMUX-ap via	Input
araddr_m0[31:0]	AXI master read addr	Input
araddr_m1[31:0]	AXI master read addr	Input
araddr_m2[31:0]	AXI master read addr	Input
araddr_m3[31:0]	AXI master read addr	Input
araddr_m4[31:0]	AXI master read addr	Input
araddr_m5[31:0]	AXI master read addr	Input
araddr_m6[31:0]	AXI master read addr	Input
araddr_m7[31:0]	AXI master read addr	Input
arburst_m0[1:0]	AXI master burst type	Input
arburst_m1[1:0]	AXI master burst type	Input
arburst_m2[1:0]	AXI master burst type	Input
arburst_m3[1:0]	AXI master burst type	Input
arburst_m4[1:0]	AXI master burst type	Input
arburst_m5[1:0]	AXI master burst type	Input
arburst_m6[1:0]	AXI master burst type	Input
arburst_m7[1:0]	AXI master burst type	Input
arcache_m0[3:0]	AXI master burst type	Input
arcache_m1[3:0]	AXI master burst type	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
arcache_m2[3:0]	AXI master burst type	Input
arcache_m3[3:0]	AXI master burst type	Input
arcache_m4[3:0]	AXI master burst type	Input
arcache_m5[3:0]	AXI master burst type	Input
arcache_m6[3:0]	AXI master burst type	Input
arcache_m7[3:0]	AXI master burst type	Input
aresetn	Main AXI reset	Input
arid_m0[3:0]	Master axi read addr id	Input
arid_m1[3:0]	Master axi read addr id	Input
arid_m2[3:0]	Master axi read addr id	Input
arid_m3[3:0]	Master axi read addr id	Input
arid_m4[3:0]	Master axi read addr id	Input
arid_m5[3:0]	Master axi read addr id	Input
arid_m6[3:0]	Master axi read addr id	Input
arid_m7[3:0]	Master axi read addr id	Input
arlen_m0[2:0]	Master axi read burst length	Input
arlen_m1[2:0]	Master axi read burst length	Input
arlen_m2[2:0]	Master axi read burst length	Input
arlen_m3[2:0]	Master axi read burst length	Input
arlen_m4[2:0]	Master axi read burst length	Input
arlen_m5[2:0]	Master axi read burst length	Input
arlen_m6[2:0]	Master axi read burst length	Input
arlen_m7[2:0]	Master axi read burst length	Input
arlock_m0[1:0]	Master axi read lock type	Input
arlock_m1[1:0]	Master axi read lock type	Input
arlock_m2[1:0]	Master axi read lock type	Input
arlock_m3[1:0]	Master axi read lock type	Input
arlock_m4[1:0]	Master axi read lock type	Input
arlock_m5[1:0]	Master axi read lock type	Input
arlock_m6[1:0]	Master axi read lock type	Input
arlock_m7[1:0]	Master axi read lock type	Input
armaster_m0[3:0]	Master ID read - m0	Input
armaster_m1[3:0]	Master ID read - m1	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
armaster_m2[3:0]	Master ID read - m2	Input
armaster_m3[3:0]	Master ID read - m3	Input
armaster_m4[3:0]	Master ID read - m4	Input
armaster_m5[3:0]	Master ID read - m5	Input
armaster_m6[3:0]	Master ID read - m6	Input
armaster_m7[3:0]	Master ID read - m6	Input
arprot_m0[2:0]	Master axi read protection type	Input
arprot_m1[2:0]	Master axi read protection type	Input
arprot_m2[2:0]	Master axi read protection type	Input
arprot_m3[2:0]	Master axi read protection type	Input
arprot_m4[2:0]	Master axi read protection type	Input
arprot_m5[2:0]	Master axi read protection type	Input
arprot_m6[2:0]	Master axi read protection type	Input
arprot_m7[2:0]	Master axi read protection type	Input
arsize_m0[1:0]	Master read burst size	Input
arsize_m1[1:0]	Master read burst size	Input
arsize_m2[1:0]	Master read burst size	Input
arsize_m3[1:0]	Master read burst size	Input
arsize_m4[1:0]	Master read burst size	Input
arsize_m5[1:0]	Master read burst size	Input
arsize_m6[1:0]	Master read burst size	Input
arsize_m7[1:0]	Master read burst size	Input
arvalid_m0	Master addr.read valid	Input
arvalid_m1	Master addr.read valid	Input
arvalid_m2	Master addr.read valid	Input
arvalid_m3	Master addr.read valid	Input
arvalid_m4	Master addr.read valid	Input
arvalid_m5	Master addr.read valid	Input
arvalid_m6	Master addr.read valid	Input
arvalid_m7	Master addr.read valid	Input
awaddr_m0[31:0]	Master axi write addr.	Input
awaddr_m1[31:0]	Master axi write addr.	Input
awaddr_m2[31:0]	Master axi write addr.	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
awaddr_m3[31:0]	Master axi write addr.	Input
awaddr_m4[31:0]	Master axi write addr.	Input
awaddr_m5[31:0]	Master axi write addr.	Input
awaddr_m6[31:0]	Master axi write addr.	Input
awaddr_m7[31:0]	Master axi write addr.	Input
awburst_m0[1:0]	Master axi write burst size	Input
awburst_m1[1:0]	Master axi write burst size	Input
awburst_m2[1:0]	Master axi write burst size	Input
awburst_m3[1:0]	Master axi write burst size	Input
awburst_m4[1:0]	Master axi write burst size	Input
awburst_m5[1:0]	Master axi write burst size	Input
awburst_m6[1:0]	Master axi write burst size	Input
awburst_m7[1:0]	Master axi write burst size	Input
awcache_m0[3:0]	Master axi write cache type	Input
awcache_m1[3:0]	Master axi write cache type	Input
awcache_m2[3:0]	Master axi write cache type	Input
awcache_m3[3:0]	Master axi write cache type	Input
awcache_m4[3:0]	Master axi write cache type	Input
awcache_m5[3:0]	Master axi write cache type	Input
awcache_m6[3:0]	Master axi write cache type	Input
awcache_m7[3:0]	Master axi write cache type	Input
awid_m0[3:0]	Master axi addr write id	Input
awid_m1[3:0]	Master axi addr write id	Input
awid_m2[3:0]	Master axi addr write id	Input
awid_m3[3:0]	Master axi addr write id	Input
awid_m4[3:0]	Master axi addr write id	Input
awid_m5[3:0]	Master axi addr write id	Input
awid_m6[3:0]	Master axi addr write id	Input
awid_m7[3:0]	Master axi addr write id	Input
awlen_m0[2:0]	Master axi write burst length	Input
awlen_m1[2:0]	Master axi write burst length	Input
awlen_m2[2:0]	Master axi write burst length	Input
awlen_m3[2:0]	Master axi write burst length	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
awlen_m4[2:0]	Master axi write burst length	Input
awlen_m5[2:0]	Master axi write burst length	Input
awlen_m6[2:0]	Master axi write burst length	Input
awlen_m7[2:0]	Master axi write burst length	Input
awlock_m0[1:0]	Master axi write lock type	Input
awlock_m1[1:0]	Master axi write lock type	Input
awlock_m2[1:0]	Master axi write lock type	Input
awlock_m3[1:0]	Master axi write lock type	Input
awlock_m4[1:0]	Master axi write lock type	Input
awlock_m5[1:0]	Master axi write lock type	Input
awlock_m6[1:0]	Master axi write lock type	Input
awlock_m7[1:0]	Master axi write lock type	Input
awmaster_m0[3:0]	Master ID write - m0	Input
awmaster_m1[3:0]	Master ID write - m1	Input
awmaster_m2[3:0]	Master ID write - m2	Input
awmaster_m3[3:0]	Master ID write - m3	Input
awmaster_m4[3:0]	Master ID write - m4	Input
awmaster_m5[3:0]	Master ID write - m5	Input
awmaster_m6[3:0]	Master ID write - m6	Input
awmaster_m7[3:0]	Master ID write - m6	Input
awprot_m0[2:0]	Master axi write protection type	Input
awprot_m1[2:0]	Master axi write protection type	Input
awprot_m2[2:0]	Master axi write protection type	Input
awprot_m3[2:0]	Master axi write protection type	Input
awprot_m4[2:0]	Master axi write protection type	Input
awprot_m5[2:0]	Master axi write protection type	Input
awprot_m6[2:0]	Master axi write protection type	Input
awprot_m7[2:0]	Master axi write protection type	Input
awsize_m0[1:0]	Master axi write burst size	Input
awsize_m1[1:0]	Master axi write burst size	Input
awsize_m2[1:0]	Master axi write burst size	Input
awsize_m3[1:0]	Master axi write burst size	Input
awsize_m4[1:0]	Master axi write burst size	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
awsize_m5[1:0]	Master axi write burst size	Input
awsize_m6[1:0]	Master axi write burst size	Input
awsize_m7[1:0]	Master axi write burst size	Input
awvalid_m0	Master axi write addr valid	Input
awvalid_m1	Master axi write addr valid	Input
awvalid_m2	Master axi write addr valid	Input
awvalid_m3	Master axi write addr valid	Input
awvalid_m4	Master axi write addr valid	Input
awvalid_m5	Master axi write addr valid	Input
awvalid_m6	Master axi write addr valid	Input
awvalid_m7	Master axi write addr valid	Input
bist_clk_en	NFC bist clock enable	Input
bit_24_cre_via	CRE on 24th weim's address bit	Input
bit_25_cre_via	CRE on 25th weim's address bit	Input
bit_26_cre_via	CRE on 26th weim's address bit	Input
bit_27_cre_via	CRE on 27th weim's address bit	Input
boot_cnfg[11:0]	weim via	Input
bp_via[4:0]	EMI IPMUX dsp via	Input
bready_m0	Master axi response ready	Input
bready_m1	Master axi response ready	Input
bready_m2	Master axi response ready	Input
bready_m3	Master axi response ready	Input
bready_m4	Master axi response ready	Input
bready_m5	Master axi response ready	Input
bready_m6	Master axi response ready	Input
bready_m7	Master axi response ready	Input
clk32	32 KHz clock in	Input
dftrst_as_528x32_b	NFC bist signal	Input
dvfs_req	emi dvfs req. from sys.	Input
dvfs_req_fast	emi dvfs req - fast arbitration	Input
dvfs_req_int2	emi dvfs req - internal2 arbitration	Input
dvfs_req_intr	emi dvfs req - internal arbitration	Input
dvfs_req_slow	emi dvfs req - slow arbitration	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
ect_out	external debug enable in the M4IF	Input
edt_clock	Test Kompress signal - added manually to EMI.v	Input
edt_update	Test Kompress signal - added manually to EMI.v	Input
edt_reset	Test Kompress signal - added manually to EMI.v	Input
edt_bypass	Test Kompress signal - added manually to EMI.v	Input
emi_iso	SRPG signals	Input
emi_pd	SRPG signals	Input
emi_pgrst	SRPG signals	Input
emi_short_b	SRPG signals	Input
EMI_spare_ports_in[49:0]	Spare Input ports	Input
endianess_m0	Master Endian	Input
endianess_m1	Master Endian	Input
endianess_m2	Master Endian	Input
endianess_m3	Master Endian	Input
endianess_m4	Master Endian	Input
endianess_m5	Master Endian	Input
endianess_m6	Master Endian	Input
endianess_m7	Master Endian	Input
esdctl_base[3:0]	ESDCTL base address	Input
esdctl_warm_reset	warm reset req. for esdctl	Input
int_mem_arready_1	internal mem read addr ready	Input
int_mem_awready_1	internal mem write addr ready	Input
int_mem_arready_2	internal2 mem read addr ready	Input
int_mem_awready_2	internal2 mem write addr ready	Input
int_mem_base_a[3:0]	internal memory base address a area	Input
int_mem_base_b[3:0]	internal memory base address b area	Input
int_mem_bid_1[3:0]	Internal Mem response id	Input
int_mem_bid_2[3:0]	Internal2 Mem response id	Input
int_mem_bresp_1[1:0]	internal mem write response	Input
int_mem_bresp_2[1:0]	internal mem write response	Input
int_mem_bvalid_1	internal mem response valid	Input
int_mem_bvalid_2	internal2 mem response valid	Input
int_mem_end_a[3:0]	internal memory end a address	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
int_mem_end_b[3:0]	internal memory end b address	Input
int_mem_rid_1[3:0]	internal mem read id 1	Input
int_mem_rid_2[3:0]	internal mem read id 2	Input
int_mem_rlast_1	internal mem read last	Input
int_mem_rvalid_1	internal mem read valid	Input
int_mem_rlast_2	internal2 mem read last	Input
int_mem_rvalid_2	internal2 mem read valid	Input
int_mem_wready_1	internal mem write ready	Input
int_mem_wready_2	internal2 mem write ready	Input
int2_mem_arready	internal2 mem read addr ready	Input
int2_mem_awready	internal2 mem write addr ready	Input
int2_mem_base[3:0]	internal memory base address area	Input
int2_mem_bid3:0]	Internal2 Mem response id	Input
int2_mem_bresp[1:0]	internal2 mem write response	Input
int2_mem_bvalid	internal2 mem response valid	Input
int2_mem_end[3:0]	internal2 memory end a address	Input
int2_mem_rid[3:0]	internal2 mem read id	Input
int2_mem_rlast	internal2 mem read last	Input
int2_mem_rvalid	internal2 mem read valid	Input
int2_mem_wready	internal2 mem write ready	Input
ipg_clk_s	IP Clock in.	Input
ipp_flash_clk	flash clock	Input
ipp_ind_dtack_b	WEIM dtack signal	Input
ipp_ind_emi_dqs[3:0]	esdctl dqs_in	Input
ipp_ind_esdctl_data[31:0]	esdctl data in	Input
ipp_ind_esdctl_scan_address[14:0]	scan signal	Input
ipp_ind_esdctl_scan_cas_b	scan signal	Input
ipp_ind_esdctl_scan_csd0	scan signal	Input
ipp_ind_esdctl_scan_csd1	scan signal	Input
ipp_ind_esdctl_scan_dqm[3:0]	scan signal	Input
ipp_ind_esdctl_scan_ras_b	scan signal	Input
ipp_ind_esdctl_scan_sdba	scan signal	Input
ipp_ind_esdctl_scan_sdcke[1:0]	scan signal	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
ipp_ind_esdctl_scan_sdclk	scan signal	Input
ipp_ind_esdctl_scan_we_b	scan signal	Input
ipp_ind_esdctl_sdclk_fb	esdctl feedback clk	Input
ipp_ind_fb_bclk	weim feedback clk	Input
ipp_ind_rdy_int	weim rdy	Input
ipp_ind_read_data[31:0]	weim input data	Input
ipp_ind_read_maddr_data[15:0]	upper 16 bits of muxed mode read data	Input
ipp_ind_wait_b	weim wait signal	Input
ipp_nfc_rb0_in	nfc NF_RB0	Input
ipp_nfc_rb1_in	nfc NF_RB1	Input
ipp_nfc_rb2_in	nfc NF_RB2	Input
ipp_nfc_rb3_in	nfc NF_RB3	Input
ipp_nfc_rb4_in	nfc NF_RB4	Input
ipp_nfc_rb5_in	nfc NF_RB5	Input
ipp_nfc_rb6_in	nfc NF_RB6	Input
ipp_nfc_rb7_in	nfc NF_RB7	Input
ipp_nfc_read_data_in[15:0]	nfc data in	Input
ipp_nfc_reset_b	NFC POR	Input
ips_addr[15:2]	IP Address	Input
ips_master_id[4:0]	Master ID for IP access	Input
ips_module_en	IP module enable	Input
ips_rwb	IP read/write command	Input
ips_wdata[31:0]	IP write data	Input
ipt_last_shift_en	enable support for last-shift	Input
ipt_mode	scan mode	Input
ipt_port_sel	scan signal to emi io muxing	Input
ipt_ram_se	NFC membist signal	Input
ipt_se	scan enable	Input
ipt_se_async	scan enable async mode	Input
ipt_se_gatedclk	scan enable gated clock	Input
ipt_si[19:0]	scan in bus	Input
lpack_int_mem_1	Low Power acknowledge from internal reg. 1	Input
lpack_int_mem_2	Low Power acknowledge from internal reg. 2	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
lpack_int2_mem	Low Power acknowledge from internal2	Input
lpmd	EMI low power mode req	Input
lpmd_fast	Fast arbit. low power mode req	Input
lpmd_int2	Internal2 mem arbit. arbit. low power mode req	Input
lpmd_intr	Internal mem arbit. arbit. low power mode req	Input
lpmd_slow	Slow arbit. low power mode req	Input
nf16boot_b	Boot from 16 bit NAND Flash	Input
nf8boot_b	Boot from 8 bit NAND Flash	Input
nf_16bit_sel	Use 8 or 16 bits Nand Flash	Input
nf_boot_with_reset	selects first command in boot sequence (reset command or read command)	Input
nfc_fms[1:0]	Flash Memory Select (512B/2KB/4KB page size)	Input
nfc_mem_pd	NFC internal RAM power down	Input
nfc_warm_reset	NFC warm reset signal	Input
rm_def_as_528x32_b[3:0]	NFC bist signal	Input
rready_m0	Master's Axi read ready.	Input
rready_m1	Master's Axi read ready.	Input
rready_m2	Master's Axi read ready.	Input
rready_m3	Master's Axi read ready.	Input
rready_m4	Master's Axi read ready.	Input
rready_m5	Master's Axi read ready.	Input
rready_m6	Master's Axi read ready.	Input
rready_m7	Master's Axi read ready.	Input
rresp_in_intr[1:0]	rresp from axi to m4if	Input
se_as_528x32_b	NFC bist signal	Input
si_as_528x32_b	NFC bist signal	Input
vl_sms_proc_emi_sms_1_capture_wr	bist signals	Input
vl_sms_proc_emi_sms_1_dm0	bist signals	Input
vl_sms_proc_emi_sms_1_dm1	bist signals	Input
vl_sms_proc_emi_sms_1_rst_sms_n	bist signals	Input
vl_sms_proc_emi_sms_1_run_bist	bist signals	Input
vl_sms_proc_emi_sms_1_select_wir	bist signals	Input
vl_sms_proc_emi_sms_1_shift_wr	bist signals	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
vl_sms_proc_emi_sms_1_smart	bist signals	Input
vl_sms_proc_emi_sms_1_update_wr	bist signals	Input
vl_sms_proc_emi_sms_1_wrck	bist signals	Input
vl_sms_proc_emi_sms_1_wrstn	bist signals	Input
vl_sms_proc_emi_sms_1_wsi	bist signals	Input
volt_chng	Voltage change input to esdctl	Input
wdata_fast[63:0]	Fast arbit. wdata bus from m4if buffers to EMI	Input
wdata_intr[63:0]	Internl arbit. wdata bus from m4if buffers to EMI	Input
wdata_slow[63:0]	write data from EMI (slow) to m4if buffers	Input
weim_nfc_base[3:0]	Slow arbitration base address	Input
weim_warm_reset	weim warm reset signal	Input
wid_m0[3:0]	Master's axi write ID	Input
wid_m1[3:0]	Master's axi write ID	Input
wid_m2[3:0]	Master's axi write ID	Input
wid_m3[3:0]	Master's axi write ID	Input
wid_m4[3:0]	Master's axi write ID	Input
wid_m5[3:0]	Master's axi write ID	Input
wid_m6[3:0]	Master's axi write ID	Input
wid_m7[3:0]	Master's axi write ID	Input
wlast_m0	Master's axi write last	Input
wlast_m1	Master's axi write last	Input
wlast_m2	Master's axi write last	Input
wlast_m3	Master's axi write last	Input
wlast_m4	Master's axi write last	Input
wlast_m5	Master's axi write last	Input
wlast_m6	Master's axi write last	Input
wlast_m7	Master's axi write last	Input
wstrb_in_int2[7:0]	Internal2 arbit. write byte strobe from m4if_buffers to EMI	Input
wstrb_in_intr[7:0]	Internal arbit. write byte strobe from m4if_buffers to EMI	Input
wstrb_intr[7:0]	internal arbit. write byte strobe from m4if_buffers to EMI	Input
wstrb_fast[7:0]	fast arbit. write byte strobe from m4if_buffers to EMI	Input
wstrb_slow[7:0]	Slow arbit. write byte strobe from m4if_buffers to EMI	Input
wvalid_m0	Master's axi wdata valid	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
wvalid_m1	Master's axi wdata valid	Input
wvalid_m2	Master's axi wdata valid	Input
wvalid_m3	Master's axi wdata valid	Input
wvalid_m4	Master's axi wdata valid	Input
wvalid_m5	Master's axi wdata valid	Input
wvalid_m6	Master's axi wdata valid	Input
wvalid_m7	Master's axi wdata valid	Input
m0_fast_bypass_via	master fast mem sync bypass	Input
m0_int2_bypass_via	master internal mem sync bypass	Input
m0_intr_bypass_via	master internal mem sync bypass	Input
m0_intrlvd_bypass_via	master interleaved bypass	Input
m0_slow_bypass_via	master slow mem sync bypass	Input
m0_wtrmrk_ap_rd	AP watermark at read	Input
m0_wtrmrk_ap_wr	AP watermark at write	Input
m0_wtrmrk_bp_rd	BP watermark at read	Input
m0_wtrmrk_bp_wr	BP watermark at write	Input
m1_fast_bypass_via	master fast mem sync bypass	Input
m1_int2_bypass_via	master internal mem sync bypass	Input
m1_intr_bypass_via	master internal mem sync bypass	Input
m1_intrlvd_bypass_via	master interleaved bypass	Input
m1_slow_bypass_via	master slow mem sync bypass	Input
m1_wtrmrk_ap_rd	AP watermark at read	Input
m1_wtrmrk_ap_wr	AP watermark at write	Input
m1_wtrmrk_bp_rd	BP watermark at read	Input
m1_wtrmrk_bp_wr	BP watermark at write	Input
m2_fast_bypass_via	master fast mem sync bypass	Input
m2_int2_bypass_via	master internal mem sync bypass	Input
m2_intr_bypass_via	master internal mem sync bypass	Input
m2_intrlvd_bypass_via	master interleaved bypass	Input
m2_slow_bypass_via	master slow mem sync bypass	Input
m2_wtrmrk_ap_rd	AP watermark at read	Input
m2_wtrmrk_ap_wr	AP watermark at write	Input
m2_wtrmrk_bp_rd	BP watermark at read	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
m2_wtrmrk_bp_wr	BP watermark at write	Input
m3_fast_bypass_via	master fast mem sync bypass	Input
m3_int2_bypass_via	master internal mem sync bypass	Input
m3_intr_bypass_via	master internal mem sync bypass	Input
m3_intrlvd_bypass_via	master interleaved bypass	Input
m3_slow_bypass_via	master slow mem sync bypass	Input
m3_wtrmrk_ap_rd	AP watermark at read	Input
m3_wtrmrk_ap_wr	AP watermark at write	Input
m3_wtrmrk_bp_rd	BP watermark at read	Input
m3_wtrmrk_bp_wr	BP watermark at write	Input
m4_fast_bypass_via	master fast mem sync bypass	Input
m4_int2_bypass_via	master internal mem sync bypass	Input
m4_intr_bypass_via	master internal mem sync bypass	Input
m4_intrlvd_bypass_via	master interleaved bypass	Input
m4_slow_bypass_via	master slow mem sync bypass	Input
m4_wtrmrk_ap_rd	AP watermark at read	Input
m4_wtrmrk_ap_wr	AP watermark at write	Input
m4_wtrmrk_bp_rd	BP watermark at read	Input
m4_wtrmrk_bp_wr	BP watermark at write	Input
m5_fast_bypass_via	master fast mem sync bypass	Input
m5_int2_bypass_via	master internal mem sync bypass	Input
m5_intr_bypass_via	master internal mem sync bypass	Input
m5_intrlvd_bypass_via	master interleaved bypass	Input
m5_slow_bypass_via	master slow mem sync bypass	Input
m5_wtrmrk_ap_rd	AP watermark at read	Input
m5_wtrmrk_ap_wr	AP watermark at write	Input
m5_wtrmrk_bp_rd	BP watermark at read	Input
m5_wtrmrk_bp_wr	BP watermark at write	Input
m6_fast_bypass_via	master fast mem sync bypass	Input
m6_int2_bypass_via	master internal mem sync bypass	Input
m6_intr_bypass_via	master internal mem sync bypass	Input
m6_intrlvd_bypass_via	master interleaved bypass	Input
m6_slow_bypass_via	master slow mem sync bypass	Input

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
m6_wtrmrk_ap_rd	AP watermark at read	Input
m6_wtrmrk_ap_wr	AP watermark at write	Input
m6_wtrmrk_bp_rd	BP watermark at read	Input
m6_wtrmrk_bp_wr	BP watermark at write	Input
m7_fast_bypass_via	master fast mem sync bypass	Input
m7_int2_bypass_via	master internal mem sync bypass	Input
m7_intr_bypass_via	master internal mem sync bypass	Input
m7_intrlvd_bypass_via	master interleaved bypass	Input
m7_slow_bypass_via	master slow mem sync bypass	Input
m7_wtrmrk_ap_rd	AP watermark at read	Input
m7_wtrmrk_ap_wr	AP watermark at write	Input
m7_wtrmrk_bp_rd	BP watermark at read	Input
m7_wtrmrk_bp_wr	BP watermark at write	Input
aclk_int2_gated	AXI internal2 memory clock - gated	Output
aclk_intr_gated	AXI internal memory clock - gated	Output
arcache_int2[3:0]	AXI arcache to internal memory	Output
arcache_intr_1[3:0]	AXI arcache to internal memory	Output
arcache_intr_2[3:0]	AXI arcache to internal memory	Output
arcache_m0[3:0]	AXI master arcache	Input
arcache_m1[3:0]	AXI master arcache	Input
arcache_m2[3:0]	AXI master arcache	Input
arcache_m3[3:0]	AXI master arcache	Input
arcache_m4[3:0]	AXI master arcache	Input
arcache_m5[3:0]	AXI master arcache	Input
arcache_m6[3:0]	AXI master arcache	Input
arcache_m7[3:0]	AXI master arcache	Input
armaster_int2[3:0]	Master ID. read - internal memory	Output
armaster_intr_1[3:0]	Master ID. read - internal memory	Output
armaster_intr_2[3:0]	Master ID. read - internal memory	Output
arready_fast	AXI arready - fast arbitration	Output
arready_m0	Master addr.read ready	Output
arready_m1	Master addr.read ready	Output
arready_m2	Master addr.read ready	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
arready_m3	Master addr.read ready	Output
arready_m4	Master addr.read ready	Output
arready_m5	Master addr.read ready	Output
arready_m6	Master addr.read ready	Output
arready_m7	Master addr.read ready	Output
arready_nfc	AXI arready of NFC	Output
arready_weim	AXI arready of WEIM	Output
awmaster_int2[3:0]	Master ID - write - internal memory	Output
awmaster_intr_1[3:0]	Master ID - write - internal memory	Output
awmaster_intr_2[3:0]	Master ID - write - internal memory	Output
awready_fast	AXI awready - fast arbitration	Output
awready_m0	Master axi write addr ready	Output
awready_m1	Master axi write addr ready	Output
awready_m2	Master axi write addr ready	Output
awready_m3	Master axi write addr ready	Output
awready_m4	Master axi write addr ready	Output
awready_m5	Master axi write addr ready	Output
awready_m6	Master axi write addr ready	Output
awready_m7	Master axi write addr ready	Output
awready_nfc	AXI awready of NFC	Output
awready_weim	AXI awready of WEIM	Output
bid_m0[3:0]	Master axi response id	Output
bid_m1[3:0]	Master axi response id	Output
bid_m2[3:0]	Master axi response id	Output
bid_m3[3:0]	Master axi response id	Output
bid_m4[3:0]	Master axi response id	Output
bid_m5[3:0]	Master axi response id	Output
bid_m6[3:0]	Master axi response id	Output
bid_m7[3:0]	Master axi response id	Output
boot_done	NFC boot done	Output
bresp_m0[1:0]	Master axi write resp.	Output
bresp_m1[1:0]	Master axi write resp.	Output
bresp_m2[1:0]	Master axi write resp.	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
bresp_m3[1:0]	Master axi write resp.	Output
bresp_m4[1:0]	Master axi write resp.	Output
bresp_m5[1:0]	Master axi write resp.	Output
bresp_m6[1:0]	Master axi write resp.	Output
bresp_m7[1:0]	Master axi write resp.	Output
bvalid_m0	Master axi write resp. valid	Output
bvalid_m1	Master axi write resp. valid	Output
bvalid_m2	Master axi write resp. valid	Output
bvalid_m3	Master axi write resp. valid	Output
bvalid_m4	Master axi write resp. valid	Output
bvalid_m5	Master axi write resp. valid	Output
bvalid_m6	Master axi write resp. valid	Output
bvalid_m7	Master axi write resp. valid	Output
chosen_data_master_fast[2:0]	Chosen master for fast arbit. data	Output
chosen_data_master_int2[2:0]	Chosen master for internal2 arbit. data	Output
chosen_data_master_intr[2:0]	Chosen master for internal arbit. data	Output
chosen_data_master_slow[2:0]	Chosen master for inter mem. arbit. data	Output
data_buf_sel_fast[5:0]	fast arbit. data buffer pointer (arbit side)	Output
data_buf_sel_int2[5:0]	slow arbit. data buffer pointer (arbit side)	Output
data_buf_sel_intr[5:0]	slow arbit. data buffer pointer (arbit side)	Output
data_buf_sel_slow[5:0]	inter. arbit. data buffer pointer (arbit side)	Output
delay_line_significant_change	ESDCTL note of delay line change	Output
dma_access0	M4IF dma access	Output
dma_access1	M4IF dma access	Output
dvfs_ack	emi dvfs ack to sys.	Output
dvfs_ack_fast	emi dvfs ack - fast arbitration	Output
dvfs_ack_int2	emi dvfs ack - internal arbitration	Output
dvfs_ack_intr	emi dvfs ack - internal arbitration	Output
dvfs_ack_slow	emi dvfs ack - slow arbitration	Output
EMI_spare_ports_out[49:0]	Spare output ports	Output
endianess_int2	Internal mem. endian	Output
endianess_intr_1	Internal mem. endian	Output
endianess_intr_2	Internal mem. endian	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
esdctl_idle	ESDCTL in idle mode, no access are served or waiting	Output
fast_clk_gate_en	fast arbitration clock - gated	Output
int_mem_araddr_1[31:0]	internal mem read addr.	Output
int_mem_arburst_1[1:0]	internal mem read burst type	Output
int_mem_arid_1[3:0]	internal mem read address id	Output
int_mem_arlen_1[2:0]	internal mem read burst length	Output
int_mem_arlock[_11:0]	internal mem read lock type	Output
int_mem_arprot_1[2:0]	internal mem read protection type	Output
int_mem_arsize_1[1:0]	internal mem read burst size	Output
int_mem_arvalid_1	internal mem read address valid	Output
int_mem_awaddr_1[31:0]	internal mem write address	Output
int_mem_awburst_1[1:0]	internal mem write burst type	Output
int_mem_awcache_1[3:0]	internal mem write cache type	Output
int_mem_awid_1[3:0]	internal mem write address id	Output
int_mem_araddr_2[31:0]	internal mem read addr.	Output
int_mem_arburst_2[1:0]	internal mem read burst type	Output
int_mem_arid_2[3:0]	internal mem read address id	Output
int_mem_arlen_2[2:0]	internal mem read burst length	Output
int_mem_arlock[_21:0]	internal mem read lock type	Output
int_mem_arprot_2[2:0]	internal mem read protection type	Output
int_mem_arsize_2[1:0]	internal mem read burst size	Output
int_mem_arvalid_2	internal mem read address valid	Output
int_mem_awaddr_2[31:0]	internal mem write address	Output
int_mem_awburst_2[1:0]	internal mem write burst type	Output
int_mem_awcache_2[3:0]	internal mem write cache type	Output
int_mem_awid_1[3:0]	internal mem write address id	Output
int_mem_awlen_1[2:0]	internal mem write burst length	Output
int_mem_awlock_1[1:0]	internal mem write lock type	Output
int_mem_awprot_1[2:0]	internal mem write protection type	Output
int_mem_awsize_1[1:0]	internal mem write burst size	Output
int_mem_awvalid_1	internal mem write address valid	Output
int_mem_bready_1	internal mem response ready	Output
int_mem_rready_1	internal mem read ready	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
int_mem_wid_1[3:0]	internal mem write id	Output
int_mem_wlast_1	internal mem write last	Output
int_mem_wvalid_intr_1	internal mem write valid	Output
int_mem_awid_2[3:0]	internal mem write address id	Output
int_mem_awlen_2[2:0]	internal mem write burst length	Output
int_mem_awlock_2[1:0]	internal mem write lock type	Output
int_mem_awprot_2[2:0]	internal mem write protection type	Output
int_mem_awsize_2[1:0]	internal mem write burst size	Output
int_mem_awvalid_2	internal mem write address valid	Output
int_mem_bready_2	internal mem response ready	Output
int_mem_rready_2	internal mem read ready	Output
int_mem_wid_2[3:0]	internal mem write id	Output
int_mem_wlast_2	internal mem write last	Output
int_mem_wvalid_intr_2	internal mem write valid	Output
int2_clk_gate_en	internal clock gating enable	Output
intr_clk_gate_en	internal clock gating enable	Output
int2_mem_araddr[31:0]	internal2 mem read addr.	Output
int2_mem_arburst[1:0]	internal2 mem read burst type	Output
int2_mem_arid[3:0]	internal2 mem read address id	Output
int2_mem_arlen[2:0]	internal2 mem read burst length	Output
int2_mem_arlock[1:0]	internal2 mem read lock type	Output
int2_mem_arprot[2:0]	internal2 mem read protection type	Output
int2_mem_arsize[1:0]	internal2 mem read burst size	Output
int2_mem_arvalid	internal2 mem read address valid	Output
int2_mem_awaddr[31:0]	internal2 mem write address	Output
int2_mem_awburst[1:0]	internal2 mem write burst type	Output
int2_mem_awcache[3:0]	internal2 mem write cache type	Output
int2_mem_awid[3:0]	internal2 mem write address id	Output
int2_mem_awlen[2:0]	internal2 mem write burst length	Output
int2_mem_awlock[1:0]	internal2 mem write lock type	Output
int2_mem_awprot[2:0]	internal2 mem write protection type	Output
int2_mem_awsize[1:0]	internal2 mem write burst size	Output
int2_mem_awvalid	internal2 mem write address valid	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
int2_mem_bready	internal2 mem response ready	Output
int2_mem_rready	internal2 mem read ready	Output
int2_mem_wid[3:0]	internal2 mem write id	Output
int2_mem_wlast	internal2 mem write last	Output
int2_mem_wvalid_intr	internal2 mem write valid	Output
ipi_emi_ap_int_b	Interrupts first OR in IPMUX	Output
ipi_emi_bp_int_b	interrupts second OR in IPMUX	Output
ipi_int_nfc_b	nfc interrupt out	Output
ipi_m4if_ap_int_b	m4if arm interrupt out	Output
ipi_m4if_bp_int_b	m4if dsp interrupt out	Output
ipi_weim_int_b	weim interrupt out	Output
ipp_do_addr_out_27_16[27:16]	WEIM addr out (bits 16-27)	Output
ipp_do_adv_b	weim adv output	Output
ipp_do_bclk	weim bclk out	Output
ipp_do_be_b[3:0]	weim be_b	Output
ipp_do_cre	weim cre bit	Output
ipp_do_cre_sel	weim cre select	Output
ipp_do_cs_b[5:0]	weim cs	Output
ipp_do_emi_debug[50:0]	debug bus	Output
ipp_do_emi_dqm[3:0]	esdctl dqm_x	Output
ipp_do_emi_dqs[3:0]	esdctl dqs	Output
ipp_do_esdctl_addr[14:0]	esdctl ma (address)	Output
ipp_do_esdctl_cas_b	esdctl cas_b	Output
ipp_do_esdctl_csd0	esdctl csd0	Output
ipp_do_esdctl_csd1	esdctl csd1	Output
ipp_do_esdctl_data[31:0]	esdctl data out	Output
ipp_do_esdctl_ras_b	esdctl ras_b	Output
ipp_do_esdctl_sdba[1:0]	esdctl ba	Output
ipp_do_esdctl_sdcke[1:0]	esdctl cke	Output
ipp_do_esdctl_sdclk	esdctl sdclk out	Output
ipp_do_esdctl_we_b	esdctl we_b	Output
ipp_do_oe_b	weim output enable	Output
ipp_do_strobe	weim strobe	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
ipp_do_we_b	weim write enable	Output
ipp_do_weim_data[31:16]	16 upper weim data bits	Output
ipp_do_weim_maddr[15:0]	lowest 16 bits of weim address	Output
ipp_do_weim_nfc_mdata_out[15:0]	weim data lowest part muxed with nfc data	Output
ipp_nfc_ale_out	fc ale out	Output
ipp_nfc_ce0_out	nfc chip select	Output
ipp_nfc_ce1_out	nfc chip select	Output
ipp_nfc_ce2_out	nfc chip select	Output
ipp_nfc_ce3_out	nfc chip select	Output
ipp_nfc_cle_out	nfc NF_CLE	Output
ipp_nfc_re_out	nfc NF_RE	Output
ipp_nfc_we_out	nfc out NF_WE	Output
ipp_nfc_wp_out	nfc out NF_WP	Output
ipp_obe_data_dir[3:2]	weim data direction for highest part	Output
ipp_obe_dqs[3:0]	esdctl dqs pads' direction	Output
ipp_obe_esdctl_data[3:0]	esdctl out enable	Output
ipp_obe_maddr_dir[1:0]	pads direction for address 0-15 for muxed mode use only	Output
ipp_obe_weim_nfc_dir[1:0]	weim data direction for lowest part muxed with nfc data direction	Output
ips_rdata[31:0]	IP read data	Output
ips_xfr_err	IP transfer error	Output
ips_xfr_wait	IP transfer wai	Output
ipt_emi_a0_in	Scan signal - address bit 0	Output
ipt_emi_a1_in	Scan signal - address bit 1	Output
ipt_emi_a2_in	Scan signal - address bit 2	Output
ipt_emi_a3_in	Scan signal - address bit 3	Output
ipt_emi_a4_in	Scan signal - address bit 4	Output
ipt_emi_a5_in	Scan signal - address bit 5	Output
ipt_emi_a6_in	Scan signal - address bit 6	Output
ipt_emi_a7_in	Scan signal - address bit 7	Output
ipt_emi_a8_in	Scan signal - address bit 8	Output
ipt_emi_a9_in	Scan signal - address bit 9	Output
ipt_emi_a10_in	Scan signal - address bit 10	Output
ipt_emi_a11_in	Scan signal - address bit 11	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
ipt_emi_a12_in	Scan signal - address bit 12	Output
ipt_emi_a13_in	Scan signal - address bit 13	Output
ipt_emi_a14_in	Scan signal - address bit 14	Output
ipt_emi_a15_in	Scan signal - address bit 15	Output
ipt_emi_d0_in	Scan signal - data bit 0	Output
ipt_emi_d1_in	Scan signal - data bit 1	Output
ipt_emi_d2_in	Scan signal - data bit 2	Output
ipt_emi_d3_in	Scan signal - data bit 3	Output
ipt_emi_d4_in	Scan signal - data bit 4	Output
ipt_emi_d5_in	Scan signal - data bit 5	Output
ipt_emi_d6_in	Scan signal - data bit 6	Output
ipt_emi_d7_in	Scan signal - data bit 7	Output
ipt_emi_d8_in	Scan signal - data bit 8	Output
ipt_emi_d9_in	Scan signal - data bit 9	Output
ipt_emi_d10_in	Scan signal - data bit 10	Output
ipt_emi_d11_in	Scan signal - data bit 11	Output
ipt_emi_d12_in	Scan signal - data bit 12	Output
ipt_emi_d13_in	Scan signal - data bit 13	Output
ipt_emi_d14_in	Scan signal - data bit 14	Output
ipt_emi_d15_in	Scan signal - data bit 15	Output
ipt_so[19:0]	scan out bus	Output
lpack	EMI low power mode ack	Output
lpack_fast	Fast arbit. low power mode ack	Output
lpack_int2	Internal2 mem. arbit. low power mode ack	Output
lpack_intr	Internal mem. arbit. low power mode ack	Output
lpack_slow	Slow arbit. low power mode ack	Output
lpmd_int_mem_1	Low Power mode to internal memory reg. 1	Output
lpmd_int_mem_2	Low Power mode to internal memoryreg. 2	Output
lpmd_int2_mem	Low Power mode to internal2 memory	Output
nfc_idle	NFC in idle mode, no access are served or waiting	Output
m0_clk_gate_en	master 0 clock gating enable signal for power saving	Output
m1_clk_gate_en	master 1 clock gating enable signal for power saving	Output
m2_clk_gate_en	master 2 clock gating enable signal for power saving	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
m3_clk_gate_en	master 3 clock gating enable signal for power saving	Output
m4_clk_gate_en	master 4 clock gating enable signal for power saving	Output
m5_clk_gate_en	master 5 clock gating enable signal for power saving	Output
m6_clk_gate_en	master 6 clock gating enable signal for power saving	Output
m7_clk_gate_en	master 7 clock gating enable signal for power saving	Output
nfc_dma_rd_req	NFC dma req.	Output
nfc_dma_wr_req	NFC dma req.	Output
one_hot_wr_sel_fast[23:0]	M4IF one hot write - fast arbitration	Output
one_hot_wr_sel_intr[23:0]	M4IF one hot write - internal arbitration	Output
one_hot_wr_sel_slow[23:0]	M4IF one hot write - slow arbitration	Output
rdata_fast[63:0]	read data from EMI to m4if buffers (fast)	Output
rdata_intr[63:0]	read data from EMI to m4if buffers (internal)	Output
rdata_sel_m0[4:0]	Master's read data buffer pointer	Output
rdata_sel_m1[4:0]	Master's read data buffer pointer	Output
rdata_sel_m2[4:0]	Master's read data buffer pointer	Output
rdata_sel_m3[4:0]	Master's read data buffer pointer	Output
rdata_sel_m4[4:0]	Master's read data buffer pointer	Output
rdata_sel_m5[4:0]	Master's read data buffer pointer	Output
rdata_sel_m6[4:0]	Master's read data buffer pointer	Output
rdata_sel_m7[4:0]	Master's read data buffer pointer	Output
rdata_slow[31:0]	read data from EMI to m4if buffers (slow)	Output
read_arbit_buf_m0[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m1[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m2[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m3[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m4[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m5[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m6[1:0]	Master's chosen buffer for read action	Output
read_arbit_buf_m7[1:0]	Master's chosen buffer for read action	Output
rid_m0[3:0]	Master's Axi read id	Output
rid_m1[3:0]	Master's Axi read id	Output
rid_m2[3:0]	Master's Axi read id	Output
rid_m3[3:0]	Master's Axi read id	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
rid_m4[3:0]	Master's Axi read id	Output
rid_m5[3:0]	Master's Axi read id	Output
rid_m6[3:0]	Master's Axi read id	Output
rid_m7[3:0]	Master's Axi read id	Output
rlast_m0	Master's Axi last read	Output
rlast_m1	Master's Axi last read	Output
rlast_m2	Master's Axi last read	Output
rlast_m3	Master's Axi last read	Output
rlast_m4	Master's Axi last read	Output
rlast_m5	Master's Axi last read	Output
rlast_m6	Master's Axi last read	Output
rlast_m7	Master's Axi last read	Output
rready_fast	AXI rready - of fast arbitrarion	Output
rready_nfc	AXI rready - of NFC	Output
rready_weim	AXI rready - of WEIM	Output
rresp_fast[1:0]	fast arbit read response to m4if buffers	Output
rresp_out_int2[1:0]	internal memory rresp to m4if_buffers	Output
rresp_out_intr[1:0]	internal memory rresp to m4if_buffers	Output
rresp_slow[1:0]	slow arbit read response to m4if buffers	Output
rvalid_buf_fast	read data valid to fast buffer	Output
rvalid_buf_int2	read data valid to internal mem buffer	Output
rvalid_buf_intr	read data valid to internal mem buffer	Output
rvalid_buf_slow	read data valid to slow buffer	Output
rvalid_m0	Master's Axi read data valid	Output
rvalid_m1	Master's Axi read data valid	Output
rvalid_m2	Master's Axi read data valid	Output
rvalid_m3	Master's Axi read data valid	Output
rvalid_m4	Master's Axi read data valid	Output
rvalid_m5	Master's Axi read data valid	Output
rvalid_m6	Master's Axi read data valid	Output
rvalid_m7	Master's Axi read data valid	Output
size_to_int2_buf[1:0]	size to read buffer	Output
size_to_intr_buf[1:0]	size to read buffer	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
slow_clk_gate_en	slow arbitration clock - gating enable	Output
so_as_528x32_b	NFC bist signal	Output
upper_to_intr_buf	M4IF to M4IF buffers	Output
vl_sms_proc_emi_sms_1_curr_err	NFC bist signal	Output
vl_sms_proc_emi_sms_1_fail_sms	NFC bist signal	Output
vl_sms_proc_emi_sms_1_ready_sms	NFC bist signal	Output
vl_sms_proc_emi_sms_1_wso	NFC bist signal	Output
vl_sms_proc_emi_sms_1_wsor	NFC bist signal	Output
wdata_sel_m0[4:0]	Master's write data buffer pointer	Output
wdata_sel_m1[4:0]	Master's write data buffer pointer	Output
wdata_sel_m2[4:0]	Master's write data buffer pointer	Output
wdata_sel_m3[4:0]	Master's write data buffer pointer	Output
wdata_sel_m4[4:0]	Master's write data buffer pointer	Output
wdata_sel_m5[4:0]	Master's write data buffer pointer	Output
wdata_sel_m6[4:0]	Master's write data buffer pointer	Output
wdata_sel_m7[4:0]	Master's write data buffer pointer	Output
weim_idle	WEIM in idle mode, no access are served or waiting	Output
wready_fast	AXI wready - of fast arbitration	Output
wready_nfc	AXI wready - of NFC	Output
wready_weim	AXI wready - of WEIM	Output
wready_m0	Master's axi write data ready	Output
wready_m1	Master's axi write data ready	Output
wready_m2	Master's axi write data ready	Output
wready_m3	Master's axi write data ready	Output
wready_m4	Master's axi write data ready	Output
wready_m5	Master's axi write data ready	Output
wready_m6	Master's axi write data ready	Output
wready_m7	Master's axi write data ready	Output
wstrb_intr_1[3:0]	Internal arbit. write strobe signal to internal mem.	Output
wstrb_intr_2[3:0]	Internal arbit. write strobe signal to internal mem.	Output
wvalid_buf_m0	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m1	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m2	Master's wdata valid from EMI to m4if buffers	Output

Table 27-1. EMI Signal Properties (continued)

Pin	Description	I/O
wvalid_buf_m3	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m4	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m5	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m6	Master's wdata valid from EMI to m4if buffers	Output
wvalid_buf_m7	Master's wdata valid from EMI to m4if buffers	Output

27.3.1 Detailed Signal Descriptions

Detailed description of any of EMI signals can be found in each of its sub module Chapter.

27.4 Memory Map and Register Definition

This section contains the EMI external/internal memory map, a register figure key, and a register summary.

27.4.1 External/Internal Memory Map

See Chapter 2, "Memory Map," to map the external and internal memories accessible by the EMI.

27.4.2 IPS Memory Map

EMI registers are mapped to IPS bus memory area and contain 16 Kbytes of space for all EMI registers. For more details on the IP registers, please refer to each EMI submodule specification.

Table 27-2 describes the IP memory space of each sub module of EMI. 0xBASE in Table 27-2 represents bits [31–14] of the IP address, whereas the rest of the address (bits [13–0]) is the EMI ips_address.

Table 27-2. EMI IP Memory Map[1]

Start ADDR	End ADDR	Module Name
0xBASE+0x0000	0xBASE+0x0FFF	M4IF
0xBASE+0x1000	0xBASE+0x1FFF	ESDCTL
0xBASE+0x2000	0xBASE+0x2FFF	WEIM
0xBASE+0x3000	0xBASE+0x3EFF	NFC
0xBASE+0x3F00	0xBASE+0x3FFF	EMI

1 0xBASE is the EMI base address as defined in the Chapter 2, "Memory Map".

27.4.3 Register Summary

The conventions in Figure 27-4 and Table 27-3 serve as a key for the register summary and individual register diagrams.

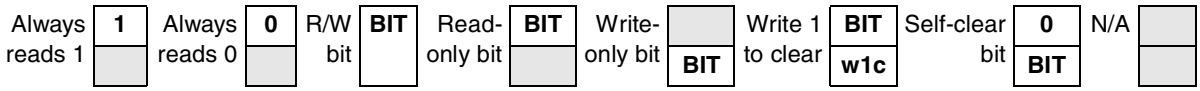


Figure 27-4. Key to M4IF Register Fields

Table 27-3 provides a key for register figures and tables and the register summary.

Table 27-3. M4IF Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 27-4 shows the M4IF register summary table.

Table 27-4. EMI Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x3F00 (IPLCK)	R																
	W																
	R								Reserved			XFR_ERR_EN	Lock_all	M4IF_lock	ESDC_lock	WEIM_lock	NFC_lock
	W																

Table 27-4. EMI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x3F04 (EICS)	R	EAOIE	EBOIE														
	W																
	R					BM4IF_AP_int	BM4IF_BP_int	BWEIM_int	BNFC_int	MAIS	MBIS	WIS	NIS	AM4IF_AP_int	AM4IF_BP_int	AWEIM_int	ANFC_int
	W																

27.4.4 Register Descriptions

This section consists of M4IF register descriptions in address order. Please refer to the next sections for detailed descriptions of each register.

27.4.4.1 IP Lock Register

IPLCK register contains the locking configuration bits of each IP registers region in EMI submodules. This register is accessible by the AP privilege master only as defined at the SoC level. Other non-privilege masters can only read the content of this register.

Address 0xBASE+0x3F00 (IPLCK) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								Reserved			XFR_ERR_EN	Lock_all	M4IF_lock	ESDC_lock	WEIM_lock	NFC_lock
W																
Reset	N/A	N/A	N/A	N/A	N/A	N/A	N/A	1	1	0	1	0	0	0	0	0

Figure 27-5. IP Lock Register Description

Detailed description of IP Lock Register is shown in Table 27-5.

Table 27-5. IP Lock Register Field Descriptions

Field	Description
31–9	N/A.
8–6	Reserved.
5 XFR_ERR_EN	EMI's ips_xfr_err enable bit. 0 ips_xfr_err is disabled (always equal to 0). 1 ips_xfr_err is enabled.

Table 27-5. IP Lock Register Field Descriptions (continued)

Field	Description
4 Lock_all	Lock all EMI registers from being configured by non privilidge masters. 0 register is unlocked. 1 register is locked.
3 M4IF_lock	Lock M4IF registers from being configured by non privilege masters (except BP privilege registers). 0 register is unlocked. 1 register is locked.
2 ESDC_lock	Lock ESDCTL registers from being configured by non privilege masters. 0 register is unlocked. 1 register is locked.
1 WEIM_lock	Lock WEIM registers from being configured by non privilege masters. 0 register is unlocked. 1 register is locked.
0 NFC_lock	Lock NFC register from being configured by non privilege masters. 0 register is unlocked. 1 register is locked.

27.4.4.2 EMI Interrupt Control and Status Register

EICS register confgured the ORed interrupt scheme. It contains a masking bit for the ORed interrupt as well. EICS is a general type register. When the Lock_all bit in IPLCK register is high, only the AP master can to write to EICS register. When the Lock_all bit is low, all other masters as well as AP can write to it.

Address 0xBASE+0x3F04 (EICS) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EAOIE	EBOIE														
W																
Reset	1	1	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					BM4IF_AP_int	BM4IF_BP_int	BWEIM_int	BNFC_int	MAIS	MBIS	WIS	NIS	AM4IF_AP_int	AM4IF_BP_int	AWEIM_int	ANFC_int
W																
Reset	N/A	N/A	N/A	N/A	0	1	1	1	0	0	0	0	1	0	1	1

Figure 27-6. EMI Interrupt Control and Status Register Description

A detailed description of the EMI Interrupt Control and Status Register is shown in Table 27-5.

Table 27-6. EMI Interrupt Control and Status Register Field Descriptions

Field	Description
31 EAOIE	EMI AP_ORed Interrupt Enable. This bit enables the AP_ORed interrupt generation of EMI. For example, if EAOIE bit is set and one of EMI sub-module interrupt is generated, AP_ORed interrupt based on its configuration is generated as well. 0 EMI AP_ORed interrupt is disabled. 1 EMI AP_ORed interrupt is enabled. (default)
30 EBOIE	EMI BP_ORed Interrupt Enable. This bit enables the BP_ORed interrupt generation of EMI. For example, if EBOIE bit is set and one of EMI sub-module interrupt is generated, BP_ORed interrupt based on its configuration is generated as well. 0 EMI BP_ORed interrupt is disabled. 1 EMI BP_ORed interrupt is enabled. (default)
29-12	Reserved.
11 BM4IF_AP_int	BP M4IF_AP_int This bit defines whether M4IF AP interrupt is included in the EMI BP_ORed interrupt or not. 0 interrupt is not included in BP_ORed.(default) 1 interrupt is included in BP_ORed.
10 BM4IF_BP_int	BP M4IF_BP_int This bit defines whether M4IF BP interrupt is included in the EMI BP_ORed interrupt or not. 0 interrupt is not included in BP_ORed. 1 interrupt is included in BP_ORed. (default)
9 BWEIM_int	BP WEIM_int This bit defines whether WEIM interrupt is included in the EMI BP_ORed interrupt or not. 0 interrupt is not included in BP_ORed. 1 interrupt is included in BP_ORed. (default)
8 BNFC_int	BP NFC_int This bit defines whether NFC interrupt is included in the EMI BP_ORed interrupt or not.. 0 interrupt is not included in BP_ORed. 1 interrupt is included in BP_ORed. (default)
7 MAIS	M4IF AP interrupt status. This bit indicates if M4IF AP interrupt had occurred. MAIS is updated in the first positive edge of ipg_clk_s in an IP read access to EICS. 0 M4IF AP interrupt did not occur. 1 M4IF AP interrupt had occurred.
6 MBIS	M4IF BP interrupt status. This bit indicates if M4IF BP interrupt had occurred. MBIS is updated in the first positive edge of ipg_clk_s in an IP read access to EICS. 0 M4IF BP interrupt did not occur. 1 M4IF BP interrupt had occurred.
5 WIS	WEIM interrupt status. This bit indicates if WEIM interrupt had occurred. WIS is updated in the first positive edge of ipg_clk_s in an IP read access to EICS. 0 WEIM interrupt did not occur. 1 WEIM interrupt had occurred.

Table 27-6. EMI Interrupt Control and Status Register Field Descriptions (continued)

Field	Description
4 NIS	NFC interrupt status. This bit indicates if NFC interrupt had occurred. NIS is updated in the first positive edge of ipg_clk_s in an IP read access to EICS. 0 NFC interrupt did not occur. 1 NFC interrupt had occurred.
3 AM4IF_AP_int	AP M4IF_AP_int This bit defines whether M4IF AP interrupt is included in the EMI AP_ORed interrupt or not. 0 interrupt is not included in AP_ORed. 1 interrupt is included in AP_ORed. (default)
2 AM4IF_BP_int	AP M4IF_BP_int This bit defines whether M4IF BP interrupt is included in the EMI AP_ORed interrupt or not. 0 interrupt is not included in AP_ORed.(default) 1 interrupt is included in AP_ORed.
1 AWEIM_int	AP WEIM_int This bit defines whether WEIM interrupt is included in the EMI AP_ORed interrupt or not. 0 interrupt is not included in AP_ORed. 1 interrupt is included in AP_ORed. (default)
0 ANFC_int	AP NFC_int This bit defines whether NFC interrupt is included in the EMI AP_ORed interrupt or not.. 0 interrupt is not included in AP_ORed. 1 interrupt is included in AP_ORed. (default)

27.5 Functional Description

As was described above, EMI is the external memory arbitration and controller, and the internal memory arbitration of the system. Each master access to the internal or external memory region would be routed out through EMI. The EMI is based on AXI pipeline separate read and write buses and therefore it can reorder the access of the masters inside the arbitration to provide the highest throughput on the external memory bus.

EMI has separate arbitration control modules for fast and slow external/internal interfaces, therefore, an access to the slow interface can be served in parallel to an access of the fast interface without any interference between them.

In addition EMI interface support asynchronous master interface, Therefore, a slow master would be able to read/write data in parallel to other fast master accesses. The arbitration logic would optimize the access between the masters to the memory controller to provide maximum throughput on the external memory bus and in the same time a minimum latency for pending access in the arbitration.

In order to provide high performance on the fast memory channel, it was decided to separate the buses of the memory controllers and to share only between several memory controllers located on the same arbitration. In EMI case, SDR and DDR SDRAM memory controller shared the same bus as well as NFC and WEIM. However, I/O pin sharing between WEIM and SDR SDRAM controller is not supported.

Many changes were done in order to reduce power in idle and low power mode. Power saving modes were added to provide automatic capability of reducing the power consumption without any special user configuration.

DVFS is supported for both frequency and voltage changes based on improved handshake mechanism between EMI and the system.

A detailed functional description on the sub module functionality can be found in each module Chapter. Please refer to module specific spec Chapter for more details.

27.5.1 Clocks

EMI contains the following clock domains:

- EMI IPS clock.
- ESDCTL main clock, up to 166 MHz
- Slow arbitration clock, up to 133 MHz
- Internal memory arbitration 1 and 2 clocks, up to 133 MHz
- 8 × master clocks that can be asynchronous to EMI arbitration clocks (66 MHz–133 MHz).
- NFC main clock, should be integer divided from slow arbitration clock. (up to 50 MHz)
- SDCLK—SDR/DDR clock to SDR/DDR SDRAM device.
- BCLK—NOR Flash/PSRAM clock WEIM in synchronous mode.

The EMI arbitrator (M4IF) assumes all of its three main clocks—ESDCTL main clock(aclk_fast) , sSlow arbitration clock (aclk_slow), and internal memory arbitration clock (aclk_intr)—are on by default after reset. If the system wishes to turn one of the clocks off, it must do so by going through the specific lpmd sequence. For example, if the system wishes to turn off "aclk_fast", it must wake up with this clock on, send an "lpmd_fast" request, wait for "lpack_fast" acknowledge, and only then turn off "aclk_fast." "lpmd_fast" must be asserted for the whole duration of "aclk_fast" being off.

27.5.2 Reset

In general EMI supports POR reset, cold reset, and warm reset.

27.5.2.1 Power On Reset

Power on reset is a reset which interfaces with the NFC memory controller. Please refer to the NFC chapter for a detailed description on POR reset functionality.

27.5.2.2 Cold Reset

Cold Reset is the general ARESET signal that enters EMI. It generates a hardware reset to all submodules of EMI. All logic is being reset when cold reset is asserted besides the POR reset logic.

27.5.2.3 Warm Reset

Warm reset is supported for the ESDCTL module only. In this mode, the data outside in the external device is kept during warm reset so that it is eady for use as soon as the device exits warm reset. Please refer to the ESDCRTL chapter for more details.

27.5.3 Interrupts

The following interrupts are supported in EMI:

- Watermark violation AP interrupt
- Watermark violation BP interrupt
- NFC interrupt
- WEIM int for OneNand and MDOC support
- Configurable AP_ORed interrupt of all EMI interrupts
- Configurable BP_ORed interrupt of all EMI interrupts

27.5.4 Endianness

EMI supports both little- and big-endian modes (LE, BE32, and BE8). The memory controllers take care on the right storage location in the memory, based on Endianness mode and memory data bus size. Endian consideration supported is byte invariant within 32 bits word only. Byte invariant within 64 bits is not allowed. A detailed description of EMI endianess can be found in Section 27.7, EMI Endianess.”

27.5.5 IPS Interface

All EMI registers are IPS memory mapped. The registers are configurable read/write via IPS bus interface only. It is recommended that the EMI IP interface be connected on the shared peripheral bus in order to allow all IP masters to configure EMI registers.

The IPS interface unit inside EMI would combined all IPS interfaces from EMI sub modules as appears in Table 27-2. The IPS IF unit would route the access to the right sub module based on ips_module_en signal and address decoding.

EMI registers uses an accessibility policy as described here below.

Based on the fact the EMI IPS interface is located on the shared bus, only the master which owns EMI IPS IF as defined in shared peripheral bus arbiter module (SPBA) can configure EMI. Please refer to SPBA module spec for more details about ownership limitations and configurations.

In addition to that, EMI decodes ips_master_id of each IP access. At the boundary level of EMI, SoC integration would predefine by via the master ID that correspond to the privilege master and by another sets of vias the BP master ID. The privilege master has the capability to lock or unlock any register from being configured by any other master but the BP Watermark registers. Therefore, the system can ensure that when there is a lock on the registers, any register will not be changed by other master than the privilege master. In a similar way, the BP master is the only master that can lock and configure BP Watermark

registers so that even the privilege master can not changed its values. In a case of a non privilege master that tries to access a lock register, EMI does not response in error, no error signal is sent back on the bus.

Please refer to Register Definition section in this document for more details on the lock mechanism.

All EMI registers are 32bit width. Only IP accesses of 32bit data width are supported.

NOTE

There is no special mechanism inside EMI to privent IP register config while other operation on AXI taking place. Therefore, system should controll IP configuration and prevent a conflict in that case.
When the access is done to a valid address location but no register is mapped to that location, an ips_xfr_err will be generated as long as the XFR_ERR_EN bit in the IPLCK register is high. If XFR_ERR_EN bit is low the ips_xfr_err signal is disabled and remains low.

27.6 Application Information

Each submodule has application notes according to its module definition and Freescale's past experience with customer needs.

NOTE

Further refining of the spec may occur in the future, as the design, modeling, and simulations are progressing.

27.7 EMI Endianess

This section provides a detailed description of the EMI module's support for the following endian modes: LE, BE8, and BE32. The chapter describes the data arrangement on the data bus in all stages inside EMI, starting with the AXI interface at M4IF toward the external memory bus at the different memory controllers.

27.7.1 AXI Endianess Support

EMI supports AXI masters with bus width of ×64 and ×32. The following sections apply to these bus width only.

27.7.2 AXI Master ×64

This section shows the data location on AXI bus as it entered into EMI on M4IF AXI gaskets. In the next section we will show the data byte locations inside EMI toward the external bus and the storage at external memory devices.

27.7.2.1 Little-Endian Mode

Figure 27-7 shows the data byte location relative to the data bus indexes in little-endian (LE) mode.

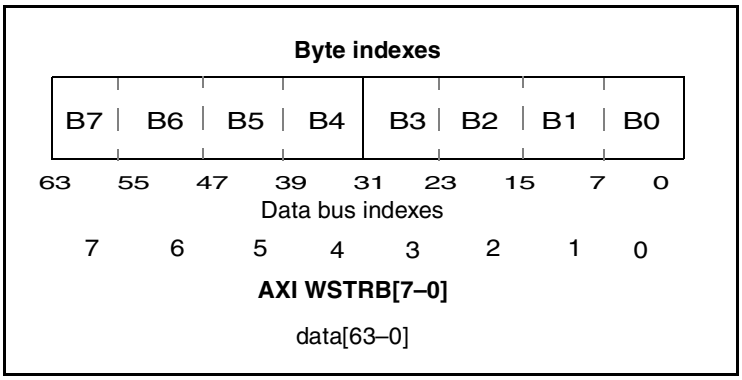


Figure 27-7. Little-Endian Byte Placement on AXI ×64

It can be seen from this illustration that the data byte with index 0 uses byte lane 0 (data[7–0]), and the data byte with index 7 uses byte lane 7 (data[63–56]).

27.7.2.2 Big-Endian Mode

The EMI module has two different big-endian modes: BE32 and BE8.

27.7.2.2.1 BE32 Mode

Figure 27-8 shows the data byte location relative to the data bus indexes in big-endian (BE32) mode.

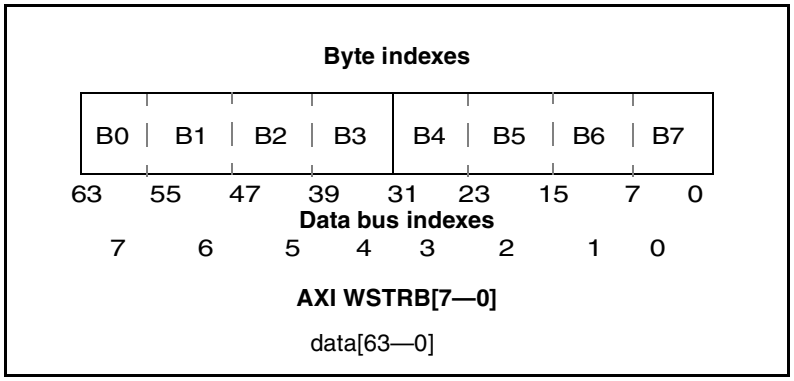


Figure 27-8. Big-Endian Byte Placement on AXI ×64 (BE32)

It can be seen from the illustration that the data byte with index 0 (least significant) uses byte lane 7 (data[63–56]), and the data byte with index 2 uses byte lane 5 (data[47–40]).

27.7.2.2.2 BE8 Mode

Figure 27-9 shows the data byte location relative to the data bus indexes in big-endian (BE8) mode.

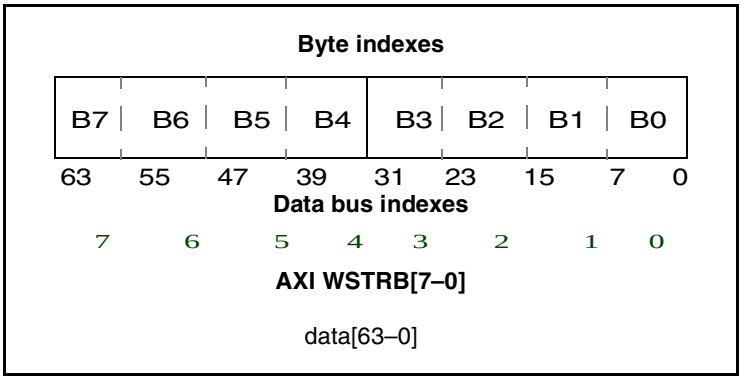


Figure 27-9. Big-Endian Byte Placement on AXI ×64 (BE8)

It can be seen from the illustration that the data byte with index 0 (least significant) uses byte lane 0 (data[7–0]), and the data byte with index 2 uses byte lane 2 (data[23–16]).This data arrangement is similar to LE mode.

The example figures below (Figure 27-10–Figure 27-13) are to show how to find which bytes are valid for each address/access size combination. Note that the example figures use the same data as their little-endian examples:

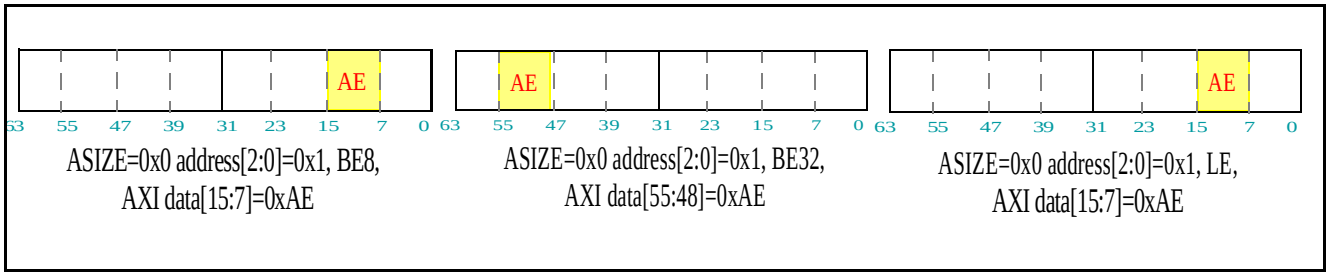


Figure 27-10. Byte Access for AXI ×64

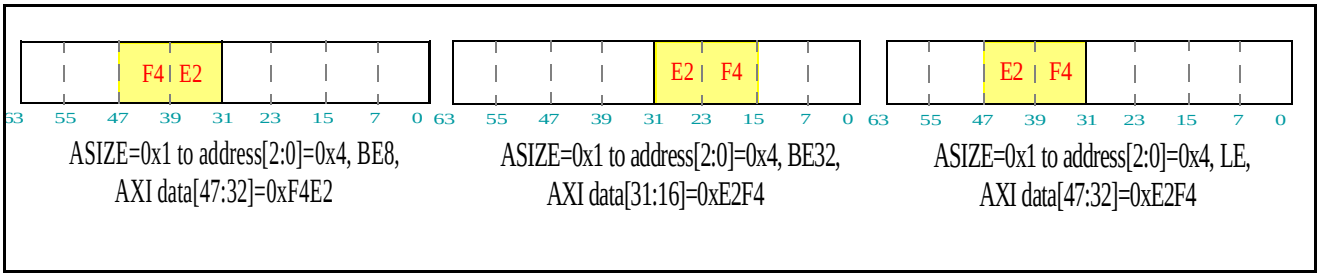


Figure 27-11. Half-Word Access for AXI ×64

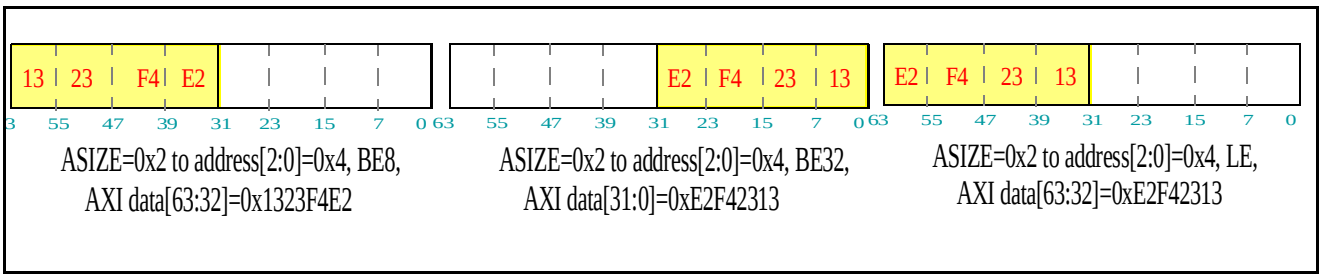


Figure 27-12. Word Access for AXI ×64

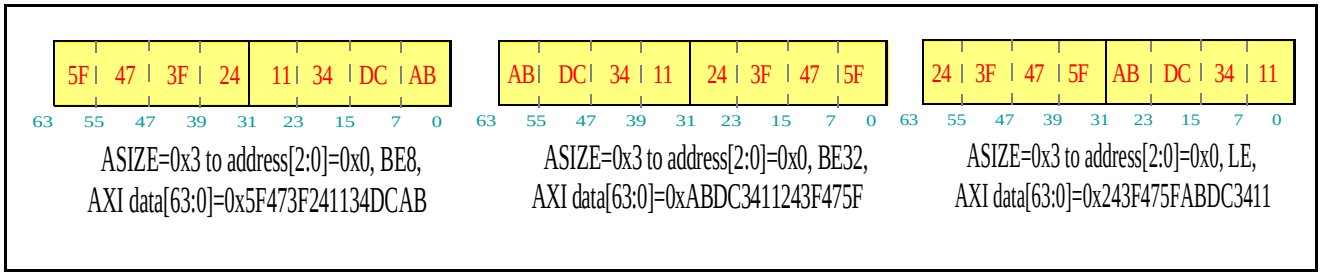


Figure 27-13. Double-Word Access for AXI ×64

27.7.3 AXI Master ×32

This section shows the data location on AXI bus as it is entered into EMI on M4IF AXI gaskets. The next section shows the data byte locations inside EMI toward the external bus and the storage at external memory devices.

27.7.3.1 Little-Endian Mode

Figure 27-14 shows the data byte location relative to the data bus indexes in little-endian mode.

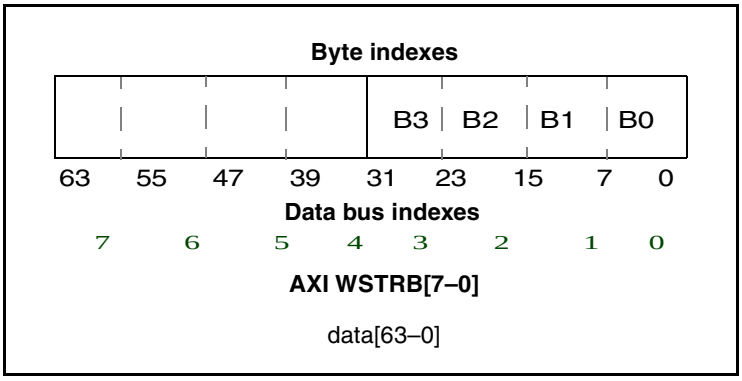


Figure 27-14. Little Endian Bytes Placement on AXI ×32

It can be seen from this illustration that the data byte with index 0 uses byte lane 0 (data[7–0]), and the data byte with index 3 uses byte lane 3 (data[31–24]).

27.7.3.2 Big-Endian Mode

The EMI module has two different big-endian modes: BE32 and BE8.

27.7.3.2.1 BE32 Mode

Figure 27-15 shows the data byte location relative to the data bus indexes in big-endian (BE32) mode.

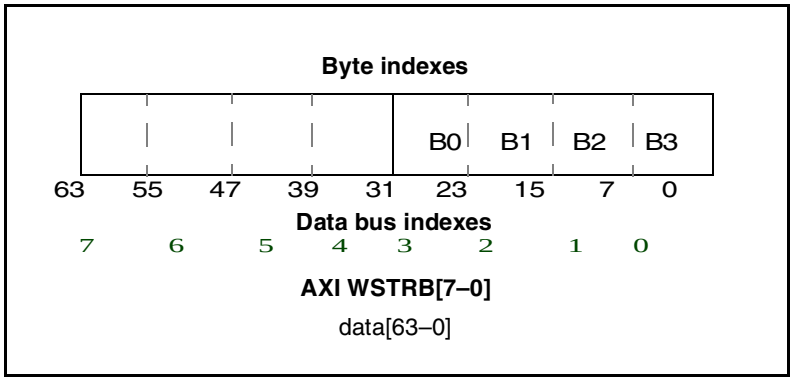


Figure 27-15. Big Endian Bytes Placement on AXI ×32 (BE32)

It can be seen from the illustration that the data byte with index 0 (least significant) uses byte lane 3 (data[31–24]), and the data byte with index 2 uses byte lane 1 (data[15–8]).

27.7.3.2.2 BE8 Mode

Figure 27-16 shows the data byte location relative to the data bus indexes in big-endian (BE8) mode.

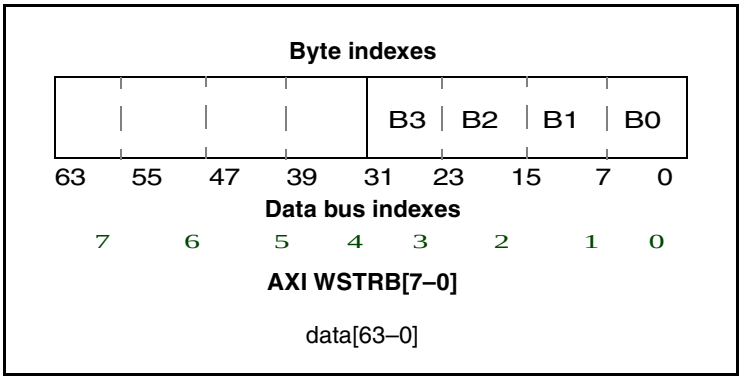


Figure 27-16. Big Endian Bytes Placement on AXI ×64 (BE8)

It can be seen from the illustration that data byte with index 0 (least significant) uses byte lane 0 (data[7–0]) and data byte with index 2 uses byte lane 2 (data[23–16]).This data arrangement is similar to LE mode.

The following examples (Figure 27-17–Figure 27-19) show the data location on AXI bus in the different modes:

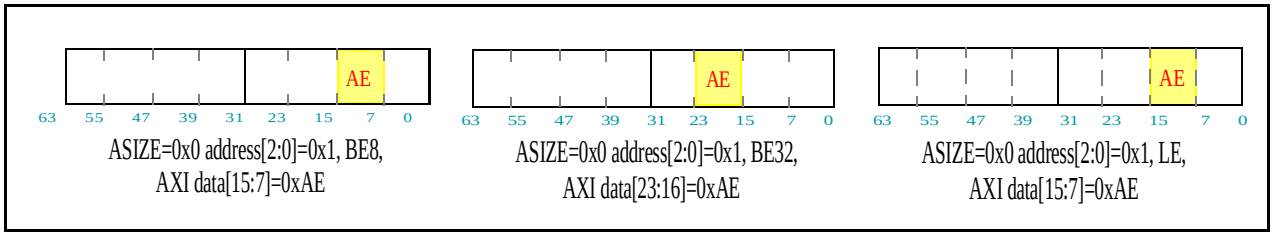


Figure 27-17. Byte Access for AXI ×32

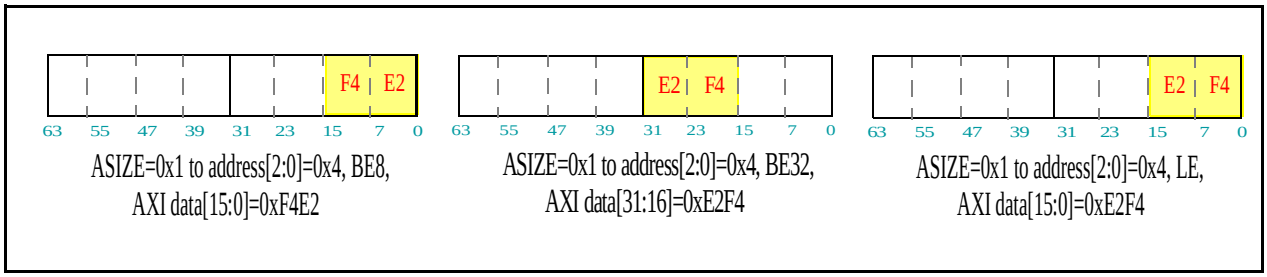


Figure 27-18. Half-Word Access for AXI ×32

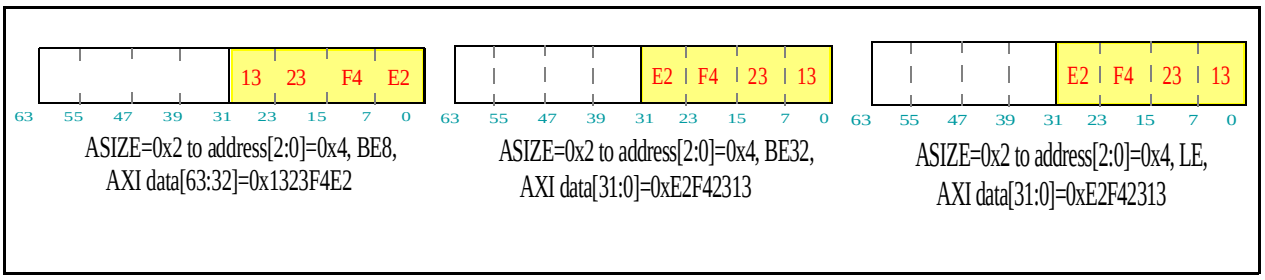


Figure 27-19. Word Access for AXI ×32

27.7.4 EMI Endianess Support

The following sections describe the endianess support in each of the EMI submodules. In general, EMI supports LE and BE32, and no special support is required for BE8.

Note that ×64 master and ×32 master can be connected to EMI. Therefore in a ×32 master case, it is required that the data bus connection be duplicated on the write bus while M4IF duplicates the data on the read bus in case of word access.

Figure 27-20 shows how a ×32 master should be connected to EMI. A ×64 master should be connected like a ×64 AXI bus regular definition.

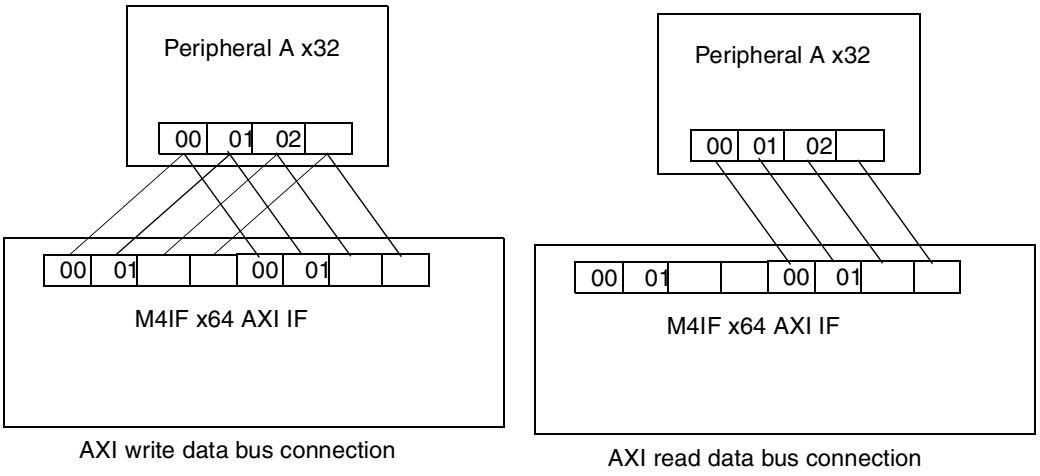


Figure 27-20. EMI Connection to AXI ×32

27.7.5 M4IF Endianess Support

This section discusses how M4IF handles the different endian modes. In general, M4IF has two different operation modes regardless of endian mode. It changes its operation logic based on fast, slow, and internal. Because the fast and internal arbitration paths are ×64 while slow arbitration is ×32, the way M4IF should provide the data to the memory controller is different.

27.7.5.1 M4IF Behavior for ×64 Arbitration

In LE or BE modes, M4IF provides the data on the bus the way it was provided from the masters in the system without any change.

27.7.5.2 M4IF Behavior for ×32 Arbitration

In LE mode, M4IF provides the data on the ×32 bus according to the byte lanes. If the access is between B0 to B3, it provides lower 32 word (byte 0–3), and when the access is between B4 to B7, it provides higher 32 word (byte 4–7).

Figure 27-21–Figure 27-23 show examples in LE mode for ×32 arbitration:

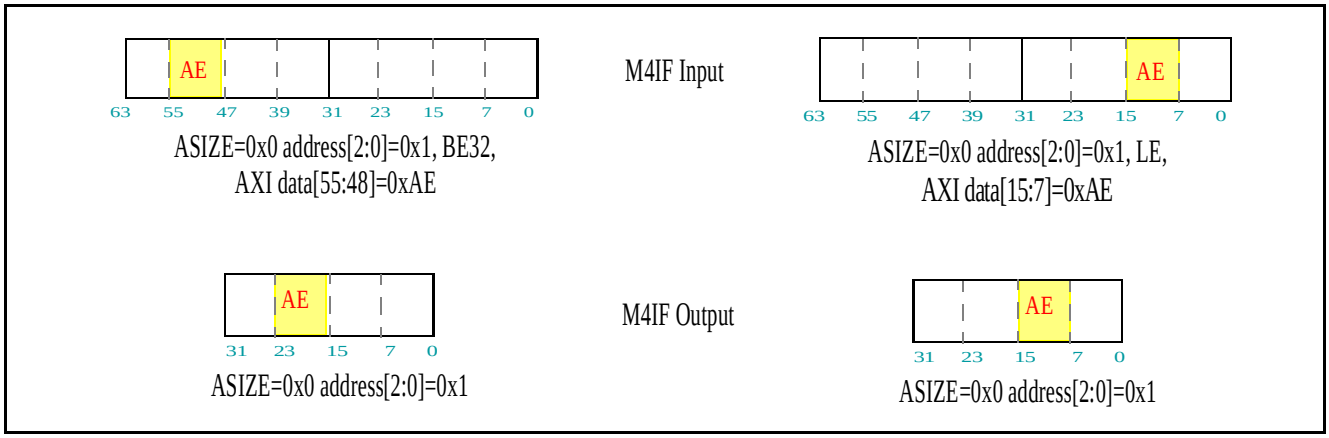


Figure 27-21. Byte Access for AXI ×64 Master and ×32 Arbitration Slave

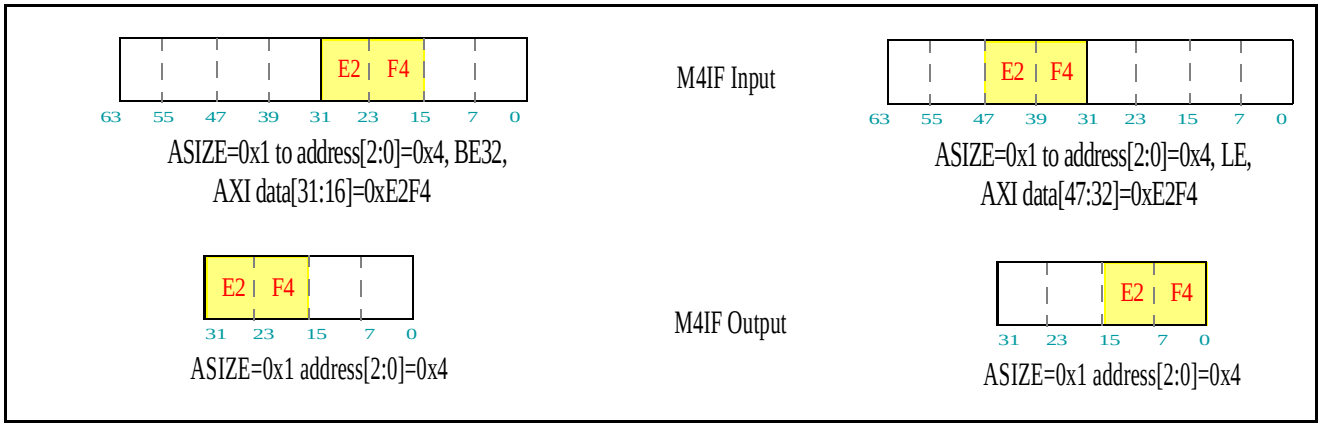


Figure 27-22. Half-Word Access for AXI ×64 Master and ×32 Arbitration Slave

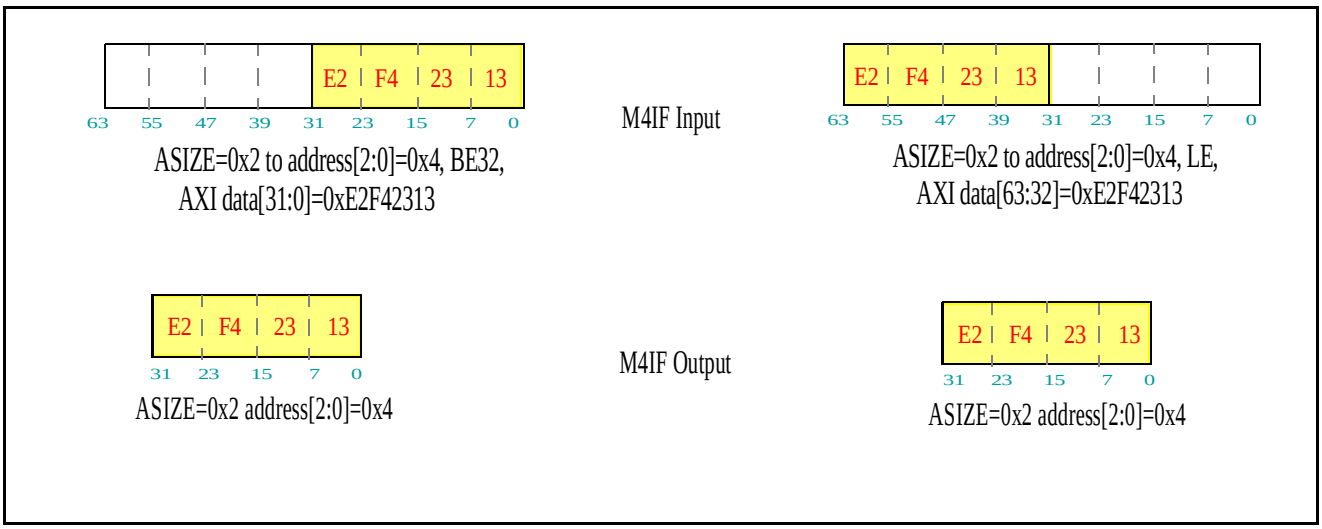


Figure 27-23. Word Access for AXI ×64 Master and ×32 Arbitration Slave

27.8 Data Storage Arrangement

This section describes how various memory controllers in EMI arrange the data in the external memory device. The section describes only ×32 and ×16 external memory devices because ×8 devices are not supported in EMI.

Figure 27-24–Figure 27-29 describe the data storage location in the external memory devices.

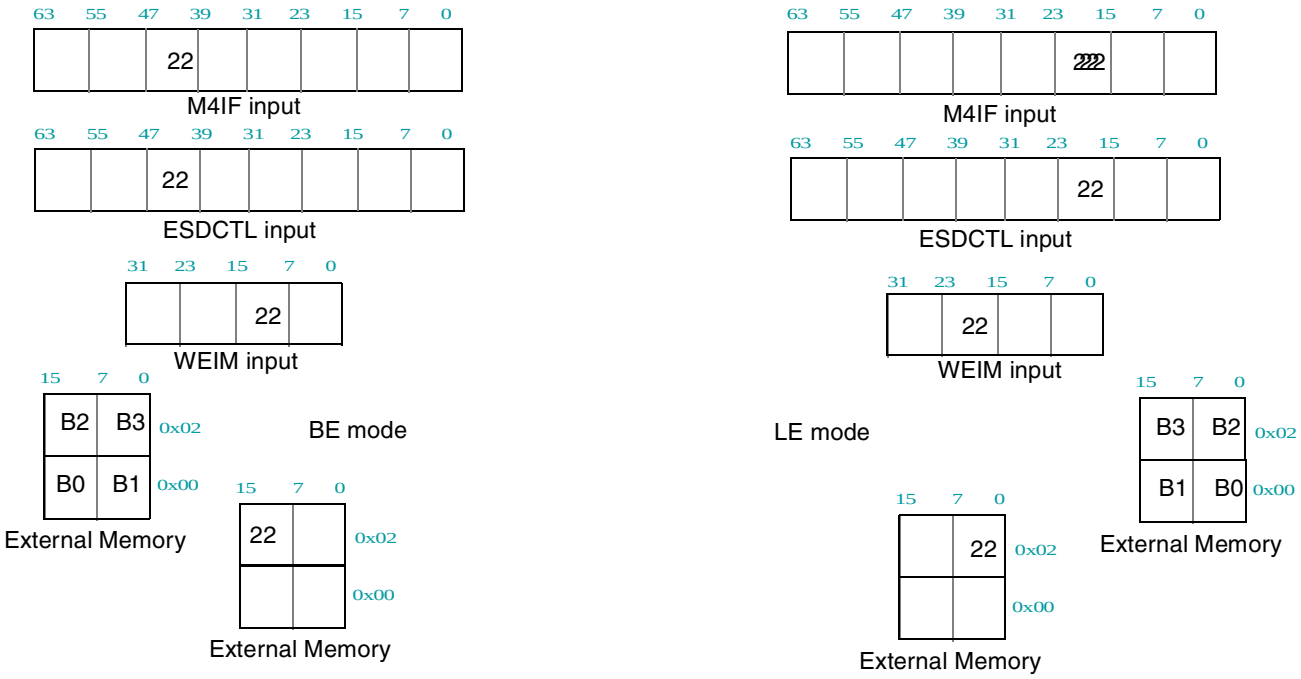


Figure 27-24. Byte Access, ASIZE = 0x0, External Memory x16

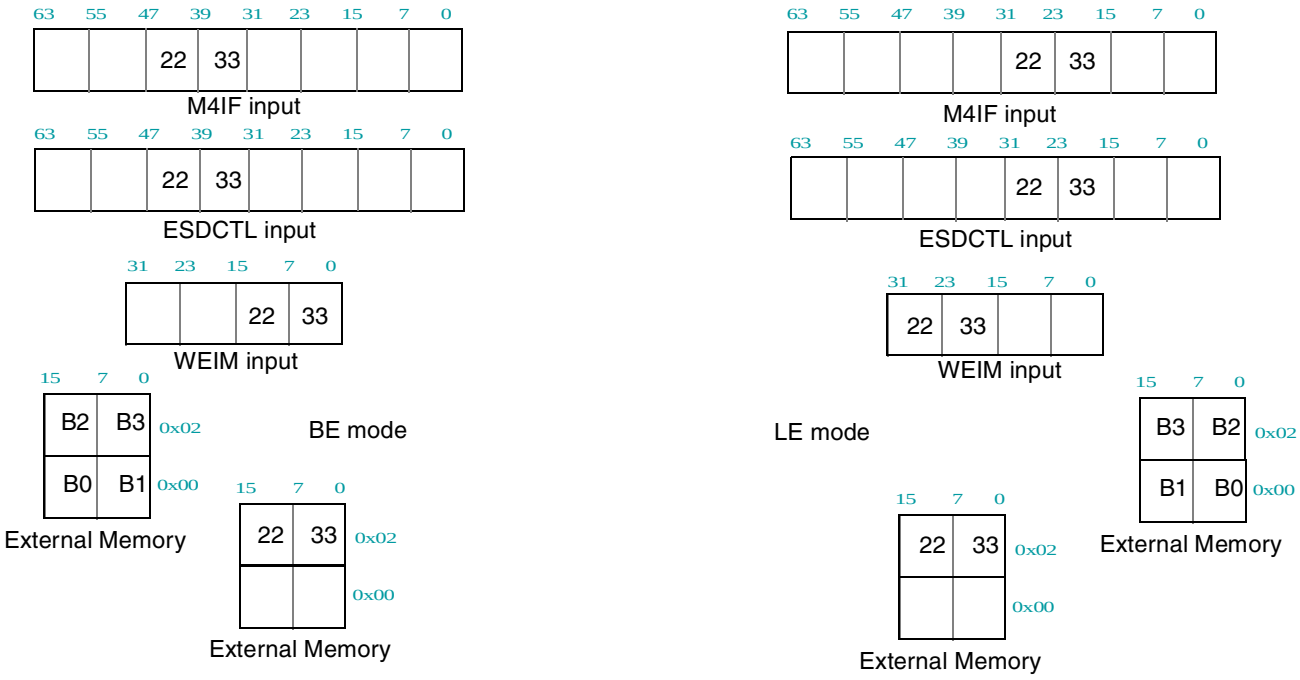


Figure 27-25. Half-Word Access, ASIZE=0x1, External Memory x16

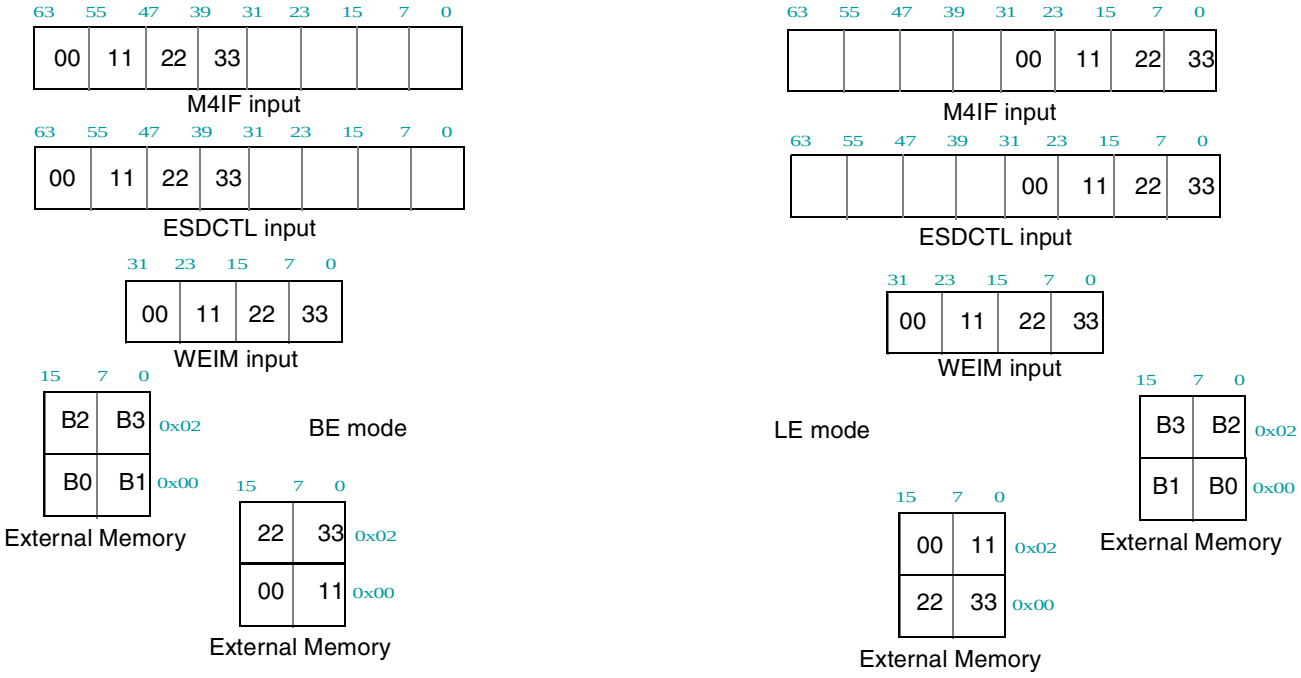


Figure 27-26. Word Access, ASIZE=0x2, External Memory x16

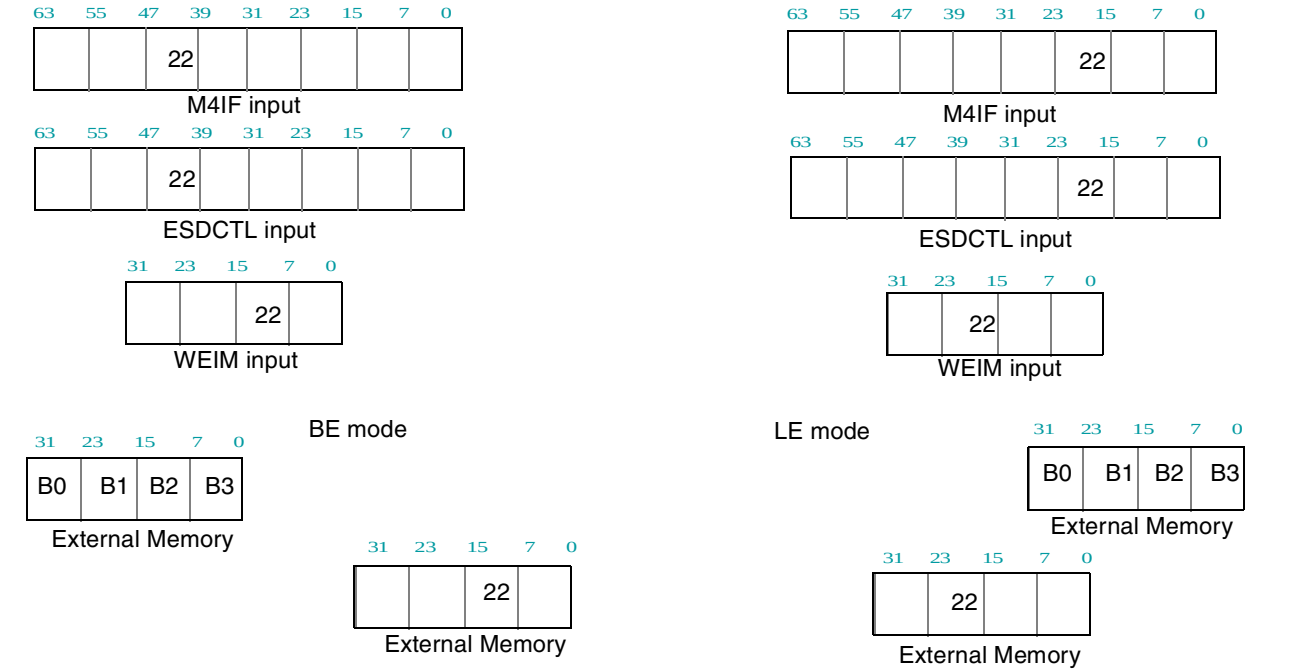


Figure 27-27. Byte Access, ASIZE=0x0, External Memory x32

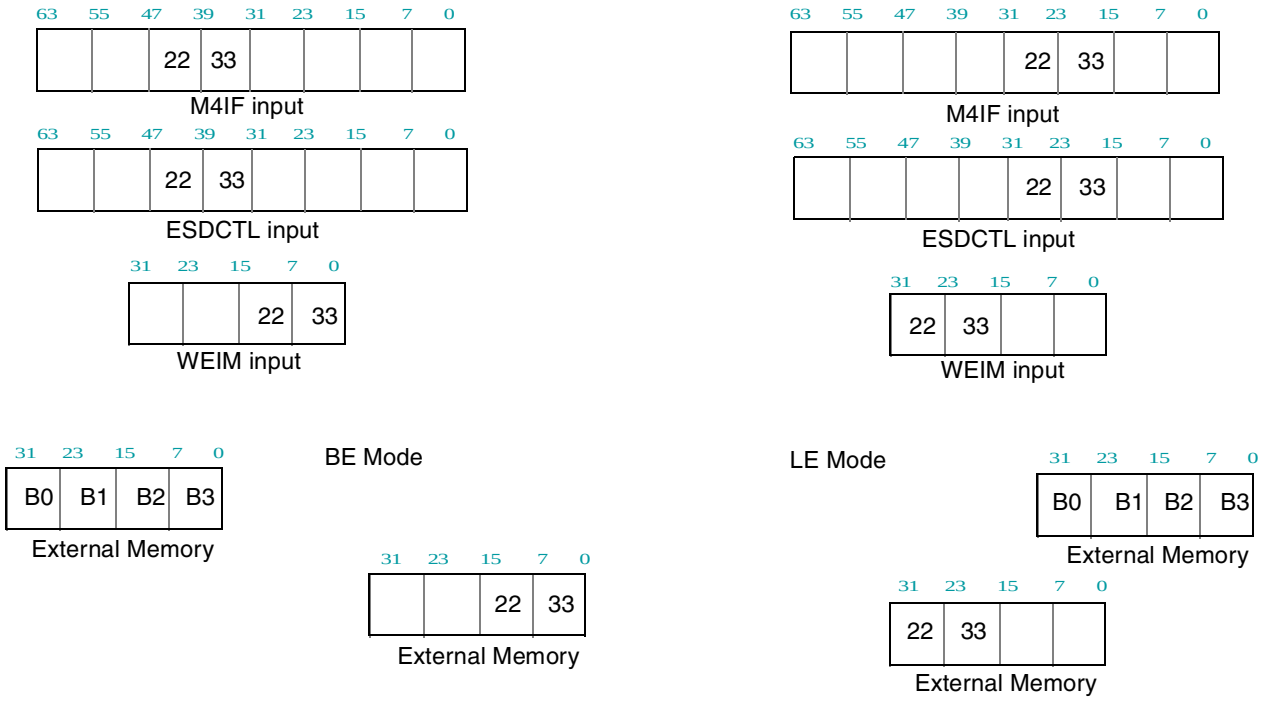


Figure 27-28. Half-Word Access, ASIZE=0x1, External Memory x32

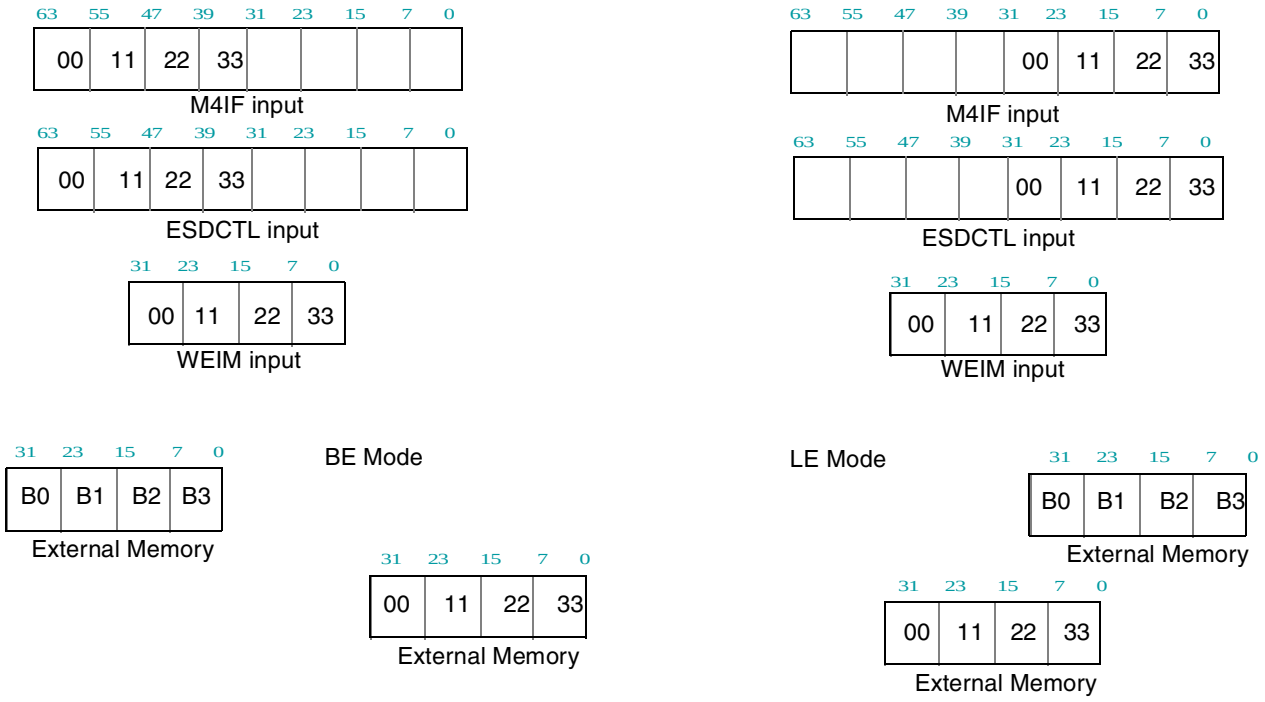


Figure 27-29. Word Access, ASIZE=0x2, External Memory x32

Chapter 28
Embedded Memory Periphery Power Gating Controller (EMPGC)

28.1 Introduction

The EMPGC controller controls the power to embedded memory periphery logic.

28.1.1 Features

The EMPG controller features include the following:

- Provide an option to switch off power to an embedded memory periphery.
- Generates power-up and power-down control sequences.
- Programmable registers to stage the power control signal.
- 32-bit IP bus interface, and all its registers are byte-accessible.

28.1.2 Modes of Operation

In functional mode, the EMPG Controller provides an option to switch off power to an embedded memory. The various power sequence control registers should be programmed to the desired value before power down.

28.2 Memory Map and Register Definition

This section provides a memory map, register key, and register summary.

28.2.1 EMPG Controller Memory Map

Table 28-1. EMPG Controller Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x000 (EMPGCR)	EMPG Control Register (EMPGCR)	R/W	0x0000_0000	28.2.3.1/28-4
0xBASE+0x004 (PUPSCR)	Power-up Sequence Control Register (PUPSCR)	R/W	0x0000_0001	28.2.3.2/28-4

Table 28-1. EMPG Controller Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x008 (PDNSCR))	Power-down Sequence Control Register (PDNSCR)	R/W	0x0000_0001	28.2.3.3/28-5
0xBASE+0x00C (EMPGSR)	EMPG Status Register (EMPGSR)	R/W	0x0000_0000	28.2.3.4/28-5

28.2.2 Register Summary

The conventions in Figure 1-1 and Table 1-7 serve as a key for the register summary and individual register diagrams.

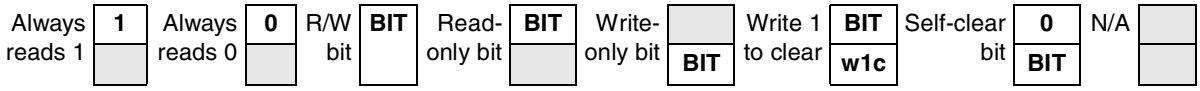


Figure 28-1. Key to Register Fields

Table 1-7 provides a key for register figures and tables and the register summary.

Table 28-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

28.2.2.1 EMPG Controller Register Summary

Table 28-3 shows the EMPG controller register summary.

Table 28-3. EMPG Controller Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (EMPGCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PCR
	W																
0xBASE+0x004 (PUPSCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	PUP							
	W																
0xBASE+0x008 (PDNSCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	PDN							
	W																
0xBASE+0x00C (EMPGSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PSR
	W																w1c

28.2.3 Register Descriptions

The EMPG Controller registers can only be accessed in supervisor mode. Any access in normal mode, or an access to an unimplemented address location, can assert transfer error signal ips_xfr_err if resp_sel is low.

28.2.3.1 EMPG Control Register (EMPGCR)

Figure 28-2 shows the EMPG control register, and Table 28-4 describes its fields.

0xBASE+0x000 (EMPGCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PCR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 28-2. EMPG Control Register (EMPGCR)

Table 28-4. Register Field Descriptions

Field	Description
0 PCR	Power control 0 Do Not Switch OFF power even if the pdn_req is asserted 1 Switch OFF Power when pdn_req is asserted WARNING: This bit should not change from “pdn_req” assertion until platform is completely powered up.

28.2.3.2 Power-up Sequence Control Register (PUPSCR)

Figure 28-3 shows the power-up sequence control register, and Table 28-5 describes its fields.

0xBASE+0x004 (PUPSCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PUP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 28-3. Power-up Sequence Control Register (PUPSCR)

Table 28-5. Register Field Descriptions

Field	Description
[7–0] PUP[7–0]	Number of clocks to de-assert PD (empgc_pd) from Power-up request. CAUTION: This should not be programmed to zero. A zero value can cause a glitch on the empgc_pd if the power-up request (pup_req) is asserted immediately after assertion of empgc_pd.

28.2.3.3 Power-down Sequence Control Register (PDNSCR)

Figure 28-4 shows the power-down sequence control register, and Table 28-6 describes its fields.

0xBASE+0x008 (PDNSCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PDN							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 28-4. Power-down Sequence Control Register (PDNSCR)

Table 28-6. Register Field Descriptions

Field	Description
[7–0] PUP[7–0]	Number of clocks from Power down request to PD (empgc_pd) assertion.

28.2.3.4 EMPG Status Register (EMPGSR)

Figure 28-5 shows the EMPG status register, and Table 28-7 describes its fields.

0xBASE+0x00C (EMPGSR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PSR
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 28-5. EMPG Status Register (EMPGSR)

Table 28-7. Register Field Descriptions

Field	Description
0 PSR	Power status. Write 1 to clear this bit. User should clear this bit after power-up, otherwise this bit will continue to reflect the power status of the very first power down. 0 was NOT powered down in previous power down request 1 was powered down in previous power down request.

28.2.4 Power-down Sequence (EMPG Entry Sequence)

Power-down sequence control register (PDNSCR) and power-up sequence control register (PUPSCR) must be programmed to the desired value before initiating the “pdn_req” and must not be changed until power up is completed. The EMPG entry sequence is outlined in the following list of steps:

1. Software program PCR bit in the EMPG control register (EMPGCR).
2. Clocks are stopped to the powering down Embedded Memory.
3. pdn_req is asserted to EMPG controller.
4. Upon detecting the pdn_req assertion, EMPG controller asserts “empgc_enable_clk”.
5. EMPG Controller asserts “empgc_pd” to isolate the outputs, based on the programming of PDNSCR[7:0].
6. EMPG controller de-asserts “empgc_enable_clk”.

28.2.4.1 Power-down Sequence Timing Waveforms

Figure 28-6 shows the power-down sequence timing waveforms.

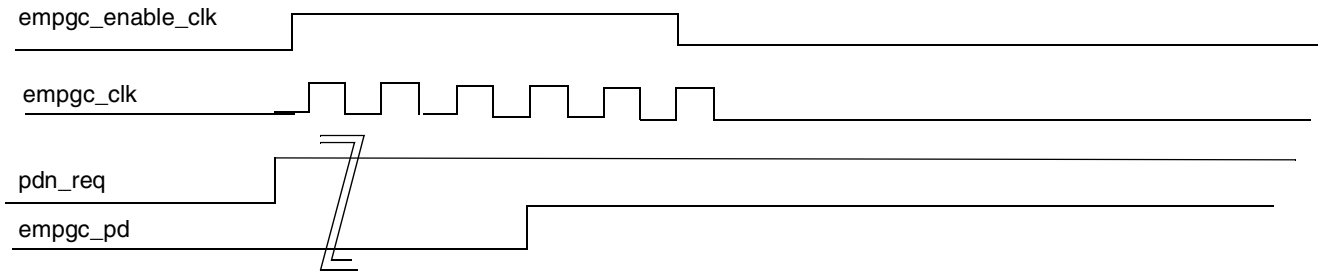


Figure 28-6. Power-down Sequence Timing Waveforms

28.2.5 Power-up Sequence (EMPG Exit Sequence)

The EMPG exit sequence is outlined in the following list of steps:

1. pup_req is asserted to EMPG Controller.
2. EMPG Controller asserts “empgc_enable_clk” to enable the clock.
3. EMPG Controller asserts “empgc_pd”, based on the programming of PUPSCR[7:0].
4. EMPG Controller de-asserts “empgc_enable_clk”.
5. “pup_req” is de-asserted to EMPG Controller.

NOTE

If pup_req is asserted before “empgc_pd” assertion then EMPG controller exits the power down sequence.

28.2.5.1 Power-up Sequence Timing Waveforms

Figure 28-7 shows the power-up sequence timing waveforms.

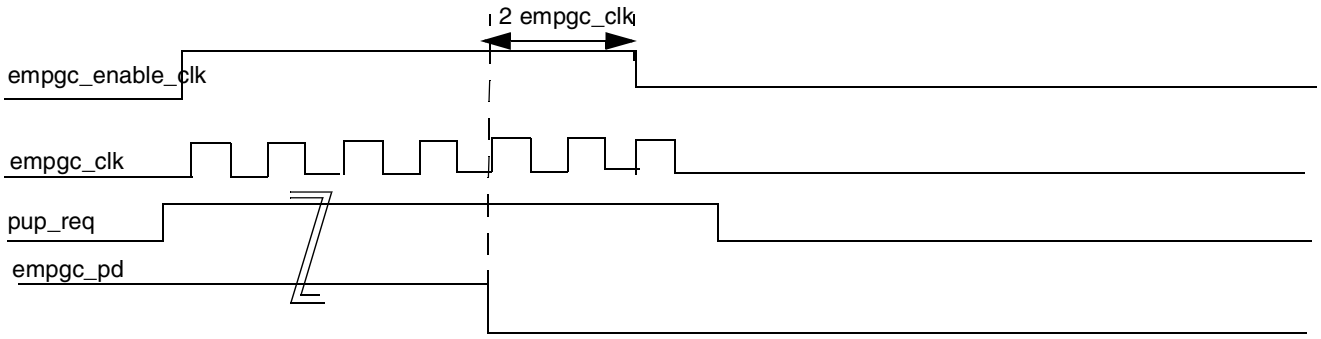


Figure 28-7. Power-up Sequence Timing Waveforms

Chapter 29
Enhanced Periodic Interrupt Timer (EPIT)

29.1 Introduction

Figure 29-1 illustrates the block diagram of the Enhanced Periodic Interrupt Timer (EPIT).

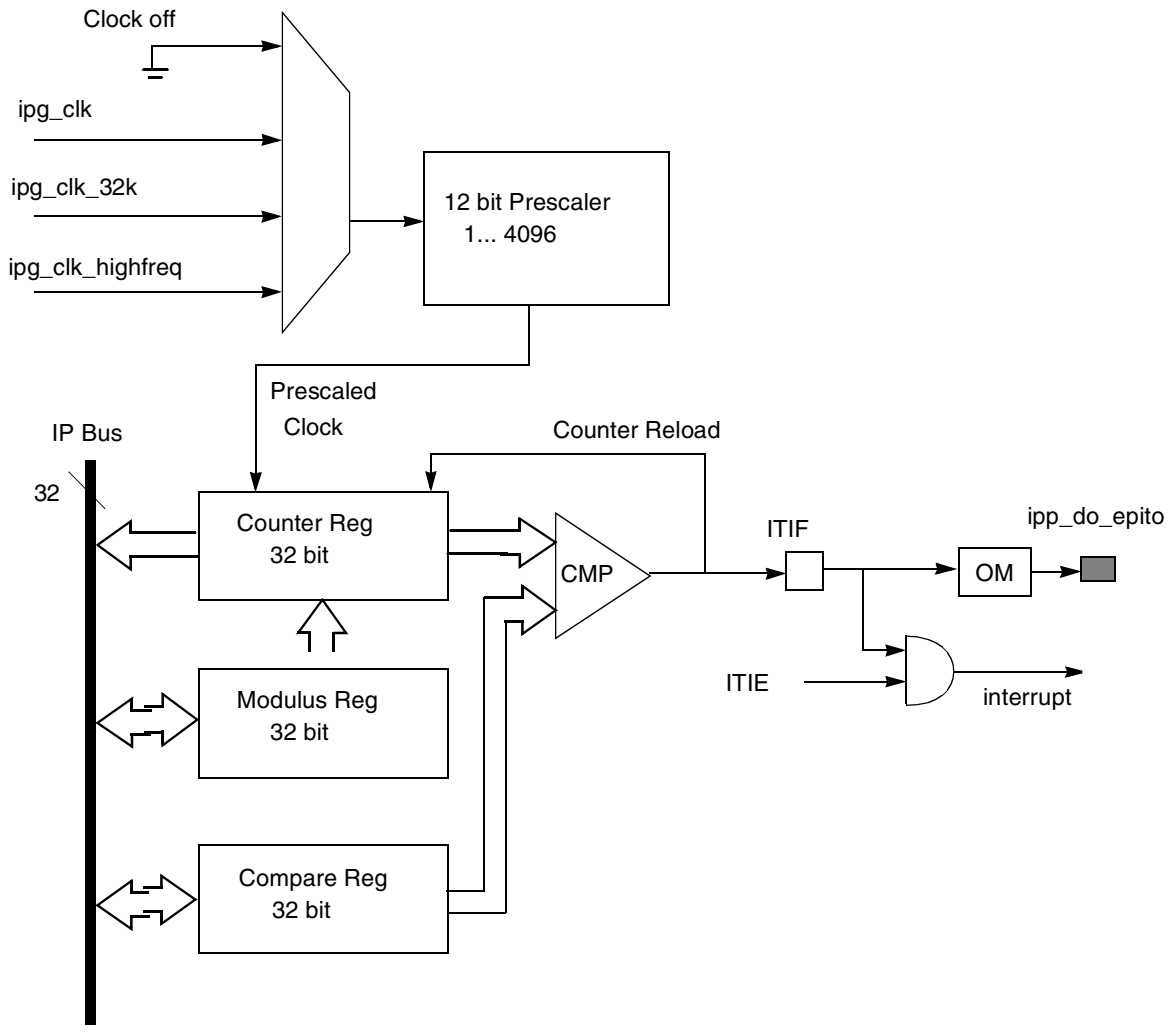


Figure 29-1. Enhanced Periodic Interrupt Timer Block Diagram

29.1.1 Overview

The Enhanced Periodic Interrupt Timer (EPIT) is a 32-bit set-and-forget timer that starts counting after the EPIT is enabled by software. It is capable of providing precise interrupts at regular intervals with minimal processor intervention.

29.1.2 Features

The following features characterize the EPIT:

- 32-bit down counter with clock source selection
- 12-bit prescaler for division of input clock frequency
- Counter value can be programmed on the fly
- Can be programmed to be active in low power and debug modes
- Interrupt generation when counter reaches the Compare value

29.1.3 Modes of Operation

The EPIT can be programmed to function in set-and-forget or free-running modes.

- Set-and-Forget mode

  This mode of operation is selected when the RLD bit in control register (EPITCR) is set to a value of one. The counter is not directly writable from the module data bus. Instead, it gets its data from the load register (EPITLR). Whenever the counter reaches a count of zero, the value in EPITLR is loaded into the counter to be decremented toward zero. The counter may by directly initialized, without having to wait for the count to reach zero, when the EPITLR is written with the IOVW bit in EPITCR set.

- Free-Running mode

  This mode of operation is selected when the RLD bit is cleared to a value of zero. In this mode, the counter rolls over from $0000_0000 to $FFFF_FFFF without reloading from the modulus register and continues counting down. In this mode, the counter can also be directly initialized by writing to EPITLR with the required initialization value after having set the IOVW bit.

29.2 External Signal Description

Table 29-1 shows the external signal properties.

Table 29-1. EPIT External Signal Properties

Name	Function	I/O	Reset	Pull Up
ipp_do_epito	Output pin at chip boundary for indication of occurrence of output compare event through a specified transition.	O	0	—

29.3 Register Definition and Memory Map

The EPIT module includes 5 user-accessible 32-bit registers. The table below summarizes these registers and their addresses.

IP Bus Write access to EPIT Control Register (EPITCR) & EPIT Load Register (EPITLR) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while other valid IP bus accesses are with 0 wait state.

Irrespective of resp_sel signal value, Write access to EPIT Counter Register EPITCNR (Read only register) will generate the bus exception (ips_xfr_err signal will be asserted). If resp_sel is driven low then Read/Write access to the unimplemented address space of EPIT (ips_addr greater than or equal to $BASE + $014) will generate the bus exception (ips_xfr_err signal will be asserted). If resp_sel is driven high then Read/Write access to the unimplemented address space of EPIT will not create any error response.

Table 29-2 shows the memory map.

Table 29-2. EPIT Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x0000 (EPITCR)	EPIT Control Register	32 bit R/W	0x0000_0000	Figure 29-3./29-5
0xBASE+0x0004 (EPITSR)	EPIT Status Register	32 bit R/W	0x0000_0000	Figure 29-4./29-7
0xBASE+0x0008 (EPITLR)	EPIT Load Register	32 bit R/W	0xFFFF_FFFF	Figure 29-5./29-8
0xBASE+0x000C (EPITCMPR)	EPIT Compare Register	32 bit R/W	0x0000_0000	Figure 29-6./29-8
0xBASE+0x0010 (EPITCNR)	EPIT Counter Register	32 bit Read	0xFFFF_FFFF	Figure 29-7./29-9

29.3.1 Register Summary

Table 29-3 summarizes the EPIT registers. Figure 29-2 provides a register key.

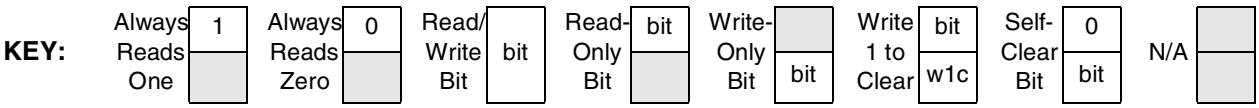


Figure 29-2. Register Key

Table 29-3. EPIT Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0004 (EPITSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OCIF
	W																w1c
0xBASE+0x0008 (EPITLR)	R	LOAD[31:16]															
	W																
	R	LOAD[15:0]															
	W																

Table 29-3. EPIT Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000C (EPITCMPR)	R	COMPARE[31:16]															
	W																
	R	COMPARE[15:0]															
	W																
0xBASE+0x0010 (EPITCNR)	R	COUNT[31:16]															
	W																
	R	COUNT[15:0]															
	W																

29.3.2 Detailed Register Descriptions

Table 29-4 shows the register terms.

Table 29-4. Register Terms

Term	Description
Grey bit	Unimplemented bit; always reads as zero; writing has no effect.
Access	—
S	Supervisor mode only
—	Supervisor or user mode
R	—
0	Always read 0.
W	—
w1c	A status bit that can be read and cleared by writing a logic 1.
Reset	—
0	Resets to a logic 0.
1	Resets to a logic 1.
u	Unaffected by reset
?	Reset state is unknown.

29.3.2.1 EPIT Control Register

The EPIT control register (EPITCR), shown in Figure 29-3, is used to configure the operating settings of the EPIT. It contains the clock division prescaler value and also the interrupt enable bit. Additionally it contains other control bit which are outlined below.

IP Bus Write access to EPIT Control Register (EPITCR) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while other valid IP bus accesses are with 0 wait state.

Address 0xBASE+0x0000 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	CLKSRC		OM		STOP EN	RES	WAIT EN	DB-GEN	IOVW	SWR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALAR[15:4]												RLD	OCIE N	EN-MOD	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 29-3. EPIT Control Register

Table 29-5 shows the EPIT control register field descriptions.

Table 29-5. Register Field Descriptions

Field	Description
31–26 Reserved	These are reserved bits and writing a value will not affect the functionality of EPIT. These reserved bits are always read as zero.
25–24 CLKSRC	Select clock source These bits determine which clock input will be selected for running the counter. This field value should only be changed when the EPIT is disabled by clearing the EN bit in this register. For other programming requirements for changing the clock source, refer to Section 29.5, Initialization/Application Information.” 00 Clock is off 01 ipg_clk 10 ipg_clk_highfreq 11 ipg_clk_32k
23–22 OM	EPIT output configuration.This bit field determines the mode of EPIT output on the output pi 00 EPIT output is disconnected from pad 01 Toggle output pin 10 Clear output pin 11 Set output pin
21 STOPEN	EPIT Stop Mode enable. This read/write control bit enables the operation of the EPIT during stop mode. This bit is reset by a hardware reset. A software reset does not affect this bit. 0 EPIT is disabled in stop mode 1 EPIT is enabled in stop mode

Table 29-5. Register Field Descriptions (continued)

Field	Description
20 RESERVED	These are reserved bits and writing a value will not affect the functionality of EPIT. These reserved bits are always read as zero.
19 WAITEN	This read/write control bit enables the operation of the EPIT during wait mode. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 EPIT is disabled in wait mode 1 EPIT is enabled in wait mode
18 DBGEN	This bit is used to keep the EPIT functional in debug mode. When this bit is cleared, the input clock is gated off in debug mode. This bit is reset by hardware reset. A software reset doesn't affect this bit. 0 Inactive in debug mode 1 Active in debug mode
17 IOVW	EPIT Counter Overwrite Enable. This bit controls the counter data when the modulus register is written. When this bit is set, all writes to the load register will overwrite the counter contents and the counter will subsequently start counting down from the programmed value. 0 Write to load register doesn't result in counter value being overwritten. 1 Write to load register results in immediate overwriting of counter value.
16 SWR	Software Reset. The EPIT module is reset when this bit is set to 1. It is a self clearing bit. This bit is set when the module is in reset state and is cleared when the reset procedure is over. Setting this bit resets all the registers to their reset values, except for the EN, ENMOD, STOPEN, WAITEN and DBGEN bits in this control register 0 EPIT is out of reset 1 EPIT is undergoing reset
15–4 PRESCALAR	Counter clock prescaler value. This bit field determines the prescaler value by which the clock will be divided before it goes to the counter 0x000 Divide by 1 0x001 Divide by 2 . . 0xfff Divide by 4096
3 RLD	Counter Reload control This bit is cleared by hardware reset. It decides the counter functionality, whether to run in "free running mode" OR "set and forget mode". 1 When the counter reaches zero it reloads from the modulus register (Set and Forget Mode) 0 When the counter reaches zero it rolls over to 0xffffffff (Free running mode)
2 OCIEN	Output Compare Interrupt Enable This bit enables the generation of interrupt on occurrence of compare event. 0 Compare interrupt disabled 1 Compare interrupt enabled

Table 29-5. Register Field Descriptions (continued)

Field	Description
1 ENMOD	EPIT Enable Mode When EPIT is disabled (EN=0), then both Main Counter and Prescaler Counter freeze their count at current count values. ENMOD bit is a r/w bit which decides the counter value when the EPIT is enabled again by setting EN bit. If ENMOD bit is set, then Main Counter is loaded with the load value (If RLD=1)/ 0xffffffff (If RLD=0) and Prescaler Counter is reset, when EPIT is enabled (EN=1). If ENMOD is programmed to 0 then both Main Counter & Prescaler Counter restart counting from their frozen values, when EPIT is enabled (EN=1). If EPIT is programmed to be disabled in a low power mode (STOP/WAIT), then both the main counter and the prescaler counter freeze at their current count values when EPIT enters low power mode. When EPIT exits the low power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 Counter will start counting from the value it had when it was disabled. 1 Counter will start count from load value (RLD=1) or 0xffffffff (If RLD=0)
0 EN	This bit enables the EPIT. EPIT Counter and Prescaler value when EPIT is enabled (EN =1), is dependent upon ENMOD & RLD bit as described for ENMOD bit. It is recommended that all registers be properly programmed before setting this bit. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 EPIT is disabled 1 EPIT is enabled

29.3.2.2 EPIT Status Register

The EPIT status register (EPITSR), shown in Figure 29-4, has a single status bit for the output compare event. It is a write 1 to clear bit.

Address 0xBASE+0x0004 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OCIF
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 29-4. EPIT Status Register

Table 29-6 shows the EPIT status register field descriptions.

Table 29-6. Register Field Descriptions

Field	Description
31–1 RESERVED	These are reserved bits and writing a value will not affect the functionality of EPIT. These reserved bits are always read as zero.
0 OCIF	Output Compare Interrupt Flag. This bit is the interrupt flag that is set when the content of counter equals the content of the compare register (EPITCMPR). This is write one to clear bit. 0 Compare event hasn't occurred 1 Compare event occurred

29.3.2.3 EPIT Load Register

The EPIT load register (EPITLR), shown in Figure 29-5, contains the value that will be loaded into the counter when EPIT counter reaches zero if the RLD bit in EPITCR is set. If the IOVW bit in the EPITCR is set then a write to this register will overwrite the value of the EPIT counter register in addition to updating this registers value. This overwrite feature is active even if the RLD bit is not set.

IP Bus Write access to EPIT load register (EPITLR) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while other valid IP bus accesses are with 0 wait state.

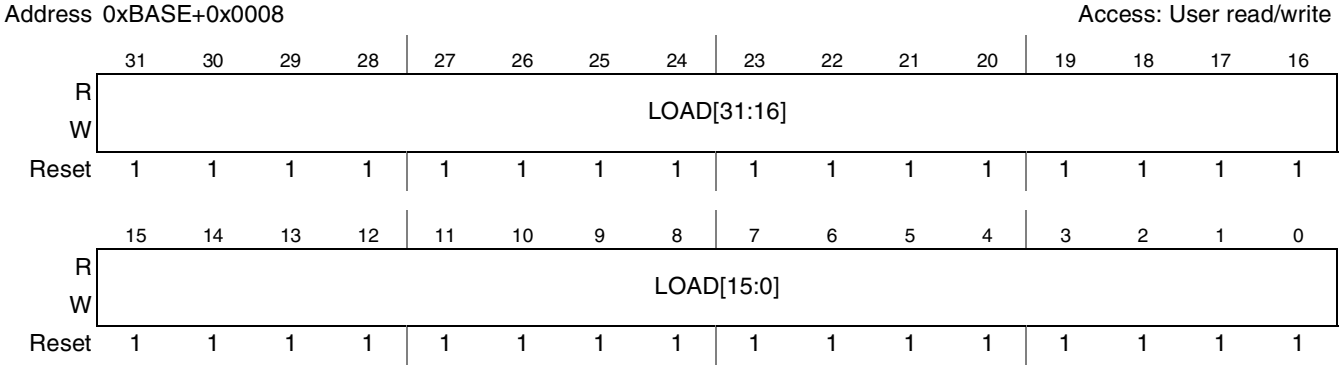


Figure 29-5. EPIT Load Register

Table 29-7 shows the EPIT load register field description.

Table 29-7. Register Field Descriptions

Field	Description
31–0 LOAD	Load Value. Value that is loaded into the counter at the start of each count cycle.

29.3.2.4 EPIT Compare Register

The EPIT compare register (EPITCMPR), shown in Figure 29-6, holds the value that determines when a compare event will be generated.

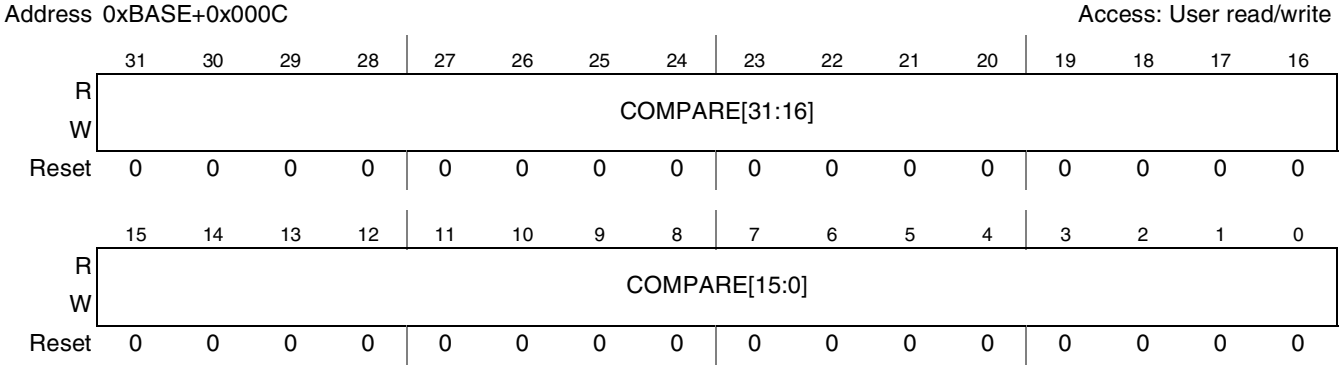


Figure 29-6. EPIT Compare Register

Table 29-8 shows the EPIT compare register field description.

Table 29-8. Register Field Descriptions

Field	Description
31–0 COMPARE	Compare Value. When the counter value equals this bit field value a compare event is generated.

29.3.2.5 EPIT Counter Register

The EPIT counter register (EPITCNR), shown in Figure 29-7, contains the current count value and can be read at any time without disturbing the counter. This is a read only register and any attempt to write into it will generate a transfer error. But if the IOVW bit in EPITCR is set, the value of this register can be overwritten with a write to EPITLR. This change is reflected when this register is read subsequently.

Address 0xBASE+0x0010 Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNT[31:16]															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT[15:0]															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 29-7. EPIT Counter Register

Table 29-9 shows the EPIT counter register field description.

Table 29-9. Register Field Descriptions

Field	Description
31–0 COUNT	Counter Value. This is the counter register value and denotes the current count state the counter register is in.

29.4 Functional Description

This section discusses the following:

- Section 29.4.1, Operation”
- Section 29.4.2, Clocks”
- Section 29.4.3, Compare Event”

29.4.1 Operation

The EPIT has a single 32-bit down counter, which starts counting when the module is enabled by software. The start value of the counter is loaded from the EPIT Load Register which can be written to at any time by the processor. The value in the compare register determines the time of occurrence of the interrupt.

When EPIT is disabled (EN = 0), then both Main Counter and Prescaler Counter freeze their count at current count values. ENMOD bit is a r/w bit which decides the counter value when the EPIT is enabled again by setting EN bit. If ENMOD bit is set, then Main Counter is loaded with the load value (If RLD = 1)/0xFFFF_FFFF (If RLD = 0) and Prescaler Counter is reset, when EPIT is enabled (EN = 1). If ENMOD is programmed to 0 then both Main Counter & Prescaler Counter restart counting from their frozen values, when EPIT is enabled (EN = 1). If EPIT is programmed to be disabled in a low power mode (STOP/WAIT), then both the main counter and the prescaler counter freeze at their current count values when EPIT enters low power mode. When EPIT exits the low power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit.

A hardware reset resets all EPIT registers to their respective reset values. There is a software reset which has the same effect on all registers except for the EN, ENMOD, STOPEN, and WAITEN bits in the control register. The state of these bits are not affected by software reset. Software reset can be asserted even when the EPIT is disabled.

29.4.2 Clocks

The clock that feeds the prescaler can be selected from among the following options:

- High frequency Clock (ipg_clk_highfreq)

  This is a high freq clock which is provided by the Clock Controller module (CCM). This clock is supposed to be on in low power mode when ipg_clk is turned off and thus EPIT can be run on this clock in low power mode. The CCM is expected to provide this clock after synchronizing it to ahb_clk in normal functional mode and switch to the unsynchronized version in low power mode.

- Low Reference Clock (ipg_clk_32k)

  This is the 32Khz low reference clock which is provided by the CCM. This clock is supposed to be on in low power mode when ipg_clk is turned off and thus EPIT can be run on this clock in low power mode. The CCM is expected to provide this clock after synchronizing it to ahb_clk in normal functional mode and switch to the unsynchronized version in low power mode.

- Global Functional Clock (ipg_clk)

  This clock is supposed to be on in normal operations, if ipg_clk is selected (CLKSRC= 01) as Clock Source. In low power modes, if EPIT is programmed to be disabled (STOPEN or WAITEN), then ipg_clk can be switched off.

The clock input source is determined by the clock source (CLKSRC) field in the control register. The clock input to the prescaler can also be disabled by programming the CLKSRC bits of control register to 00. **This field value should be changed only after disabling the EPIT by clearing the EN bit in the EPITCR**. For other programming requirements while changing clock source, refer to Section 29.5, Initialization/Application Information.”

The PRESCALER field is used to select the divide ratio of the input clock that drives the main counter. The prescaler can divide the input clock by a value between 1 and 4096. A change in the value of the PRESCALER field is immediately reflected on its output clock frequency.

Figure 29-8 shows the prescaler value change diagram.

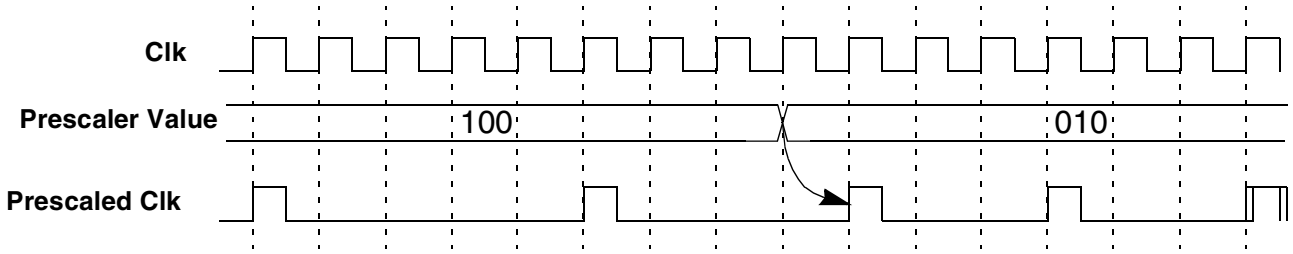


Figure 29-8. Prescaler Value Change Diagram

Whenever ipg_clk is not selected, the ipg_enable_clk is de asserted to turnoff ipg_clk to EPIT.

29.4.3 Compare Event

When the programmed content of EPITCMPR matches the value in EPITCNR, a compare status flag is set and an interrupt is generated if the corresponding bit is set in the control register. Consequently, the compare output pin will be set, cleared, toggled or not affected at all according to how the mode bits are programmed. If an interrupt is required at rollover (when counter value reaches 0x0000_0000 and new value is loaded) compare register value should be made equal to load register value in set-and-forget mode or equal to 0xFFFF_FFFF in free-running mode.

Figure 29-9 shows the compare event and interrupt timer diagram.

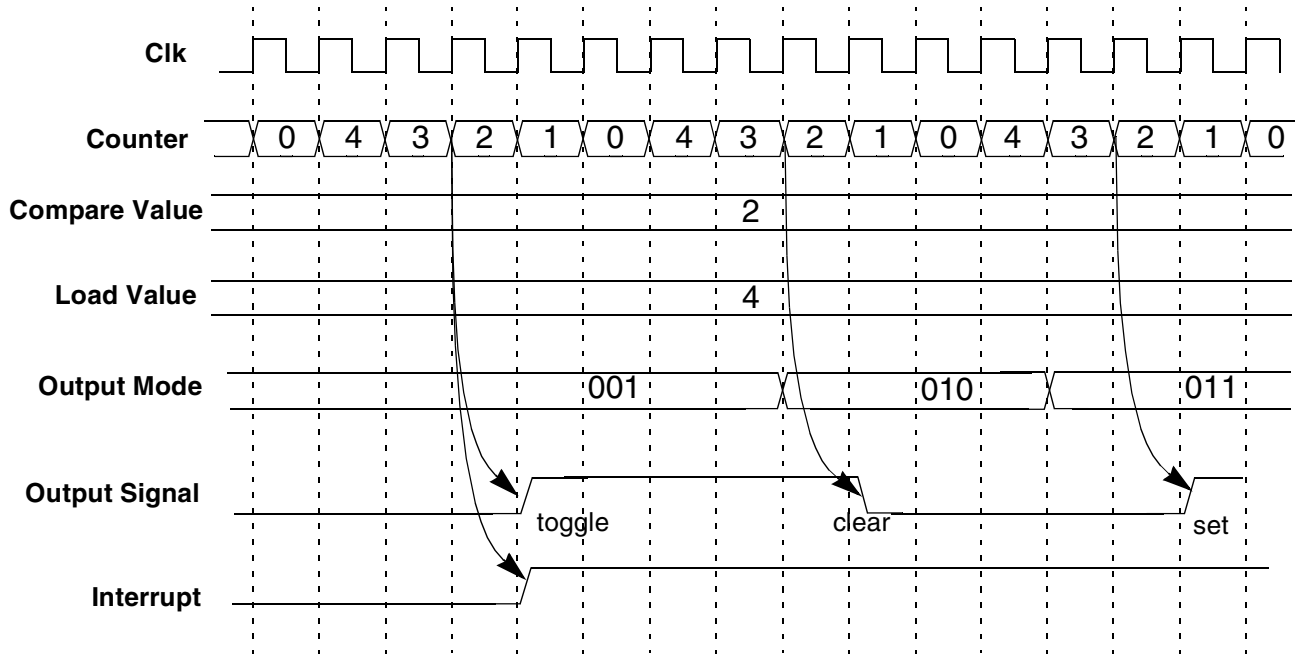


Figure 29-9. Compare Event and Interrupt Timing Diagram

29.4.3.1 Counter Value Overwrite

The EPIT counter value can be overwritten to acquire a desired value at any point of time. The procedure for this is to set the IOVW bit in the control register and then write the desired value into the load register. This will result in the load register acquiring that value and also the counter being overwritten with it. If the EPIT is running the counter resumes counting from the overwritten value.

29.4.3.2 Low Power Mode Behavior

The EPIT timer's behavior in low power modes depends on the clock source that it runs with. If the selected clock source is available and the corresponding low power enable bit is set, then the EPIT continues to function in the low power mode. If EPIT is programmed to be disabled in a low power mode (STOP/WAIT), then main counter and the prescaler counter freeze at the current count values when EPIT enters low power mode. When EPIT exits the low power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit.

Care should be taken that in the low power mode, any register read/ write operations should only be done if the selected clock running the counter is synchronous to ipg_clk_s. If the clock selected is not available or the enable bit is not set then the counter value freezes at its current value and resumes counting once the low power mode is excited.

29.4.3.3 Debug Mode Behavior

In debug mode, the EPIT timers have the option of continuing to run or be halted. If the DBGEN bit is reset in the EPIT control Register, the timer is halted. When debug mode is exited, the timer operation reverts to what it was prior to entering debug mode.

29.5 Initialization/Application Information

The CLKSRC field in EPITCR determines the clock source. This field value should be changed only after disabling the EPIT (EN = 0). The following software sequence must be used when changing the clock source.

1. Disable EPIT by setting EN = 0 in EPITCR.
2. Program OM = 00 in EPITCR.
3. Disable EPIT interrupts.
4. Program CLKSRC to desired clock source in EPITCR.
5. Clear EPIT status register (EPITSR) i.e. (w1c).
6. Enable EPIT interrupts.
7. Set ENMOD = 1 in EPITCR, to bring EPIT Counter to defined state (EPITLR value or 0xFFFF_FFFF).
8. Enable EPIT (EN = 1) in EPITCR.

Chapter 30
Enhanced SDRAM Controller (eSDCTL)

30.1 Introduction

ESDCTL is a DDR controller used for controlling DDR2 and mDDR memories. The controller sub blocks are as follows:

- Address Decoder—Used for interpreting the AXI address to the used device domain (chip select, bank, row and column)
- Config Register—For configuring the controller registers through IP accesses and storing registers values.
- Bank Model—Stores memory banks status.
- Size Logic—Used for breaking an incoming access to the required set of accesses to the memory device.
- Command Sequencer—Used to decide which command is required next from the memory.
- Command Control—Execute the chosen command when it can and control the process each command execution initiation.
- LPDDR Interface—Contain the Delay Lines and special logic for working with DDR devices which send/receive data in posedge and negedge.
- Data Path—Sort the outgoing data in Write and the incoming data in Read.

30.2 Block Diagram

Figure 30-1 shows the block diagram.

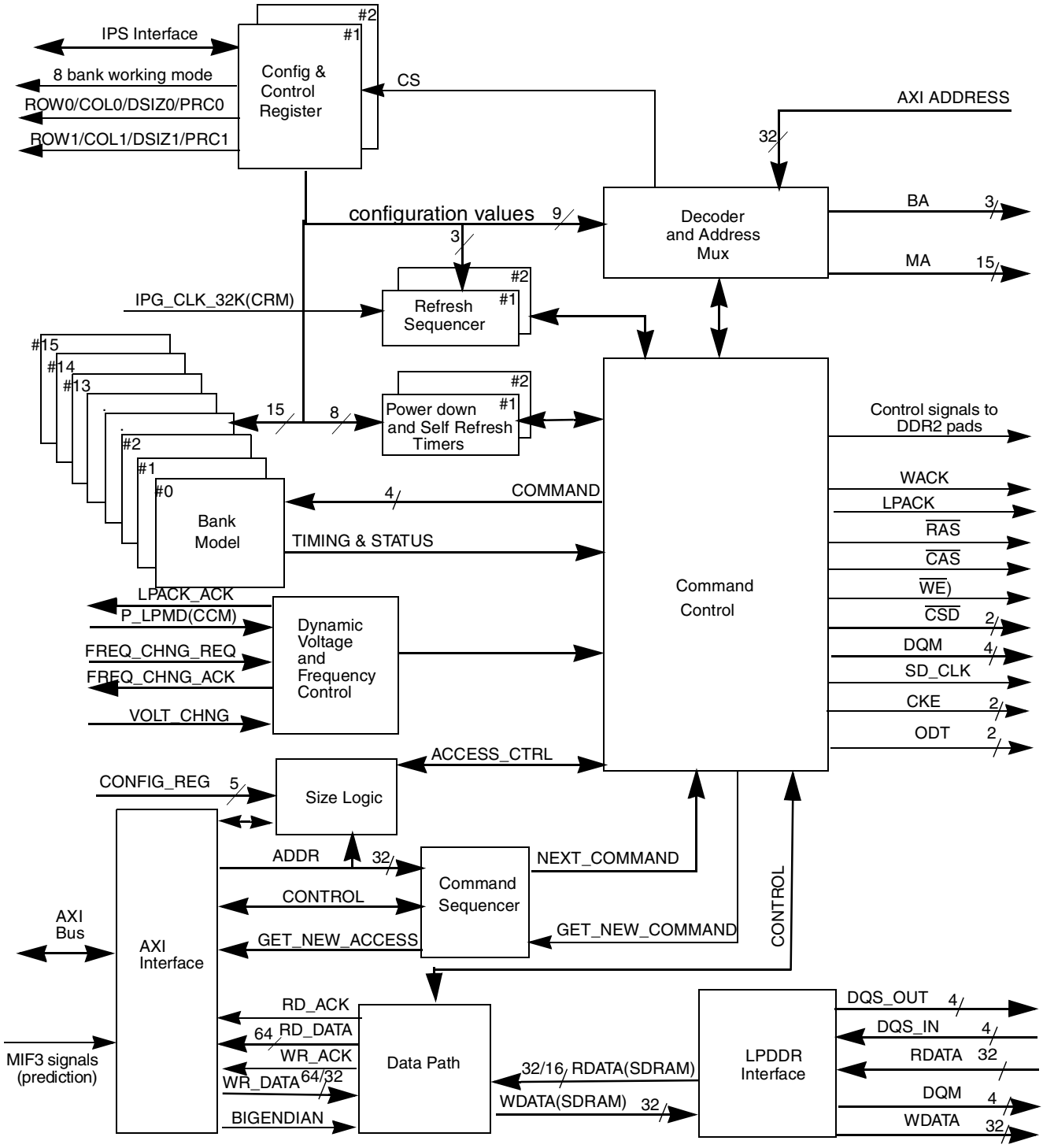


Figure 30-1. Enhanced LPDDR/DDR2 SDRAM Controller Block Diagram

30.3 Overview

The Enhanced SDRAM Controller consists of nine major blocks, including the SDRAM command state machine controller, bank register (page and bank address comparators), row/column address multiplexer, configuration registers, refresh request counter, command sequencer, size logic (splitting access), data path (data aligner/multiplexer), LPDDR interface, and the Power Down timer.

30.3.1 Features

The ESDRAMC includes the following features:

- Full support of Dynamic frequency change and voltage change for both MDDR and DDR2 modes.
  - Negotiation for frequency change with the system.
  - Constant delay line re-measuring for voltage change is supported to allow voltage change.
- Optimization of consecutive memory accesses through memory command anticipation (latency hiding)
  - Hiding latency by optimization the commands to both CS connected-command anticipation.
  - Preparation of valid waiting AXI access, in terms of ACT,PRE commands while serving another access.
  - Preparation of predicted next miss access, in term of ACT, PRE commands while serving another access.
  - Keeping track of open memory pages
  - Bank-wise memory address mapping
  - MDDR burst length configuration of 8.
  - DDR2 burst length configuration of 4.
  - AXI bus compliant
  - Shared Address and command bus to memory devices.
- Supports 64-Mbyte, 128-Mbyte, 256-Mbyte, 512-Mbyte, 4-bank, 8-bank, LPDDR or DDR2.
  - Two independent chip selects.
  - Up to 256-Mbyte per chip select.
  - Up to eight banks active simultaneously per chip select for DDR2.
  - Up to four banks active simultaneously per chip select for MDDR.
  - Support only CAS 3 devices.
  - JEDEC standard pinout/operation.
- Support for ×16/×32 mDDR/DD2 DDR400 devices.
- Support on die termination (ODT) for DDR2 devices.
- Support both differential DQS and single ended DQS modes.
- Software configurable for different system and memory devices requirements.
  - x16/x32 memory data bus width.
  - number of row and column addresses.
  - row cycle delay (t_{RC})

- auto-refresh cycle delay (t_{RFC})
- row precharge delay (t_{RP})
- row to column delay (t_{RCD})
- load mode register to active command (t_{MRD})
- write to precharge (t_{WR})
- write to read (t_{WTR}) for LPDDR memories only
- LPDDR exit power down to next valid command delay (t_{XP})
- active to precharge (t_{RAS})
- active to active (t_{RRD})
- Exit self refresh to any command (t_{XSR}/t_{XSRD} / t_{SRFX})

- Built in auto-refresh timer and state machine
- Hardware and software supported self-refresh entry and exit
  - Keeps data valid during system reset and low power modes
  - Auto Power Down timer (one per Chip Select)
  - Automatic Self Refresh timer (one per Chip Select)
  - Auto Precharge timer (one per bank in each chip select)
- Supports Deep power down enter and exit in software for LPDDR.
- Supports AXI bursts of up to 8
  - Supports bus transfers of 32 and 64 bit.
  - 8 and 16 bit transfers are supported for single accesses only.
  - Fixed burst type is not supported, other then bursts of length 1.
- Page boundary crossing support, up to 4-Kbyte boundary in AXI address.
  - Automatically generates precharge and active the next row when crossing page boundary on the AXI bus.

30.3.2 AXI Interface

The AXI interface interfaces between any AXI bus master and the ESDCTL controller. The ESDCTL supports the following features and restrictions:

Supports one AXI ×64 bus width.

- Supports clock frequency of up to 200 MHz for the AXI port.
- Out of order accesses are not supported. That means that data interleaving of two different IDs is not allowed in write cycles.
- The controller will not provide data interleaving in the AXI read data bus.
- Exclusive AXI accesses are not supported by the ESDCTL. Exclusive AXI accesses should be handled by the ESDCTL bus master.
- Atomic/Locked access are ignored. Atomic/Locked access should be handeled by the ESDCTL bus master.

- All accesses are treated as non-bufferable and non-cachable accesses. Bufferable and cachable accesses should be handeled by the ESDCTL bus master.
- The ID bus is 8 bit long.
- Bursts of up to 8 double-words (64 bits), will be supported, for incr and wrap burst types. AWSIZE/ARSIZE are 2 bits wide. AWLEN/ARLEN are 3 bits wide.
- ARRAEDY and AWREADY are all defaulted to high in idle state. Accepting a read or a write will lower the other ready as well until the controller is ready for another address for latency hiding. If both a read and a write are valid at the same cycle, the read will be serviced first, then the write, and only after the write is done the readies will go back up.
- The controller BRESP and RRESP signals are returning OKAY by default, even for exclusive access (and not EXOKAY). For accesses that are out of the memory space a DECERR will be returned, and for an access to a chip select which is not enabled, a SLVERR will be issued on bus.

NOTE

M4IF is used as ESDCTL arbirtation to buffer and manage incoming AXI accesses. Please refer to M4IF spec for more detail.

30.4 External Signal Description

This section discusses input and output signals between the Enhanced SDRAM Controller and the external memory devices. Other than the chip select outputs ($\overline{\text{CSD}}$0 and $\overline{\text{CSD}}$1), on die termination outputs (ODT0 and ODT1) and clock enables (CKE0 and CKE1), all signals are shared between the two chip select regions.

Table 30-1 summarizes the interface signals.

Table 30-1. ESDRAMC Signal Properties

Name	Function	Reset State	Direction
warm_reset	Warm synchronious Reset to ESDCTL	0	Input
reset_b	Cold Asynchronious Reset to ESDCTL	0	Input
sdclk_out	Clock to SDRAM (up to 200MHz)	1	Output
cke[0]	Clock enable to CS0	0	Output
cke[1]	Clock enable to CS1	0	Output
cs_b[0]	Chip select to CS0	1	Output
cs_b[1]	Chip select to CS1	1	Output
rd_data[31:0]	Read Data from memories	0	Input
wr_data[31:0]	Write data to memories	0	Output
out_en[3:0]	Output buffer enable for DQ pad. (bit per byte)	0	Output
ma[14:0]	Col/row addresses	0	Output
ba[2:0]	Bank address	0	Output

Table 30-1. ESDRAMC Signal Properties (continued)

Name	Function	Reset State	Direction
dqm_x[3:0]	Data Qualifier Mask (bit per byte)	0	Output
we_b	Write Enable	1	Output
ras_b	Row Address Strobe	1	Output
cas_b	Column Address Strobe	1	Output
lpack	Low Power Mode Acknowledge	1	Output
awbigend	Big endian signal for the current write transaction.	0	Input
arbigend	Big endian signal for the current read transaction.	0	Input
row0[2:0]	Number of rows as was programed for chip select number 0	0	Output
col0[1:0]	Number of columns as was programed for chip select number 0	0	Output
dsiz0[1:0]	SDRAM bus width as was programed for chip select number 0	0	Output
prc0[7:0]	Precharge bank indication for 8 banks of CS0.	0	Output
row1[2:0]	Number of rows as was programed for chip select number 1	0	Output
col1[1:0]	Number of columns as was programed for chip select number 1	0	Output
dsiz1[1:0]	SDRAM bus width as was programed for chip select number 1	0	Output
prc1[7:0]	Precharge bank indication for 8 banks of CS1.	0	Output
ddr2_8_bank	Number of banks in device(4 or 8).	0	Output
dvfs_req	Frequency change Request	0	Output
dvfs_grant	Frequency change Acknowledge	0	Input
p_lpmd	Low Power Mode Entry Request	0	Input
bi_on	Address Interleaving indication to arbiter	0	Output
volt_chng	Voltage change indication.	0	Input
freerun_clk	Free Running clock - not gated by lpmd	0	Input
clk32	32KHz clock	0	Input
emi_pd	Power down indication for SRPG cells	0	Input
sdram_base_via[3:0]	4 msb bits of CS0 in memory map.	0	Input
MIF3 signals			
mif3_p1_valid	Valid rd/wr access exist.	0	Input
mif3_p1_cs	Chip select of valid access	0	Input

Table 30-1. ESDRAMC Signal Properties (continued)

Name	Function	Reset State	Direction
mif3_p1_ba[2:0]	Bank of valid access	0	Input
mif3_p1_row[14:0]	Row of valid access	0	Input
mif3_p2_valid	Next rd/wr miss access	0	Input
mif3_p2_cs	Chip select of next miss access	0	Input
mif3_p2_ba[2:0]	Bank of next miss access	0	Input
mif3_p2_row[14:0]	Row of next miss access	0	Input
mif3_p2_ack	Bank/row of next miss access is now open, and the access is hit.	0	Output
DDR2 signals			
odt[0]	On die termination to CS0	0	Output
odt[1]	On die termination to CS1	0	Output
term_ctl0[1:0]	Termination control to DQS[0]/DQ[31:24] pads	0	Output
term_ctl1[1:0]	Termination control to DQS[1]/DQ[23:16] pads	0	Output
term_ctl2[1:0]	Termination control to DQS[2]/DQ[15:8] pads	0	Output
term_ctl3[1:0]	Termination control to DQS[3]/DQ[7:0] pads	0	Output
diff_dqs_en	Differential DQS indication to DQS[3:0] pads	0	Output
Mobile LPDDR signals			
dqs_out3	Data strobe byte 3 (D[31:24])	0	Output
dqs_out2	Data strobe byte 2 (D[23:16])	0	Output
dqs_out1	Data strobe byte 1 (D[15:8])	0	Output
dqs_out0	Data strobe byte 0 (D[7:0])	0	Output
dqs_in3	Data strobe byte 3 (D[31:24])	0	Input
dqs_in2	Data strobe byte 2 (D[23:16])	0	Input
dqs_in1	Data strobe byte 1 (D[15:8])	0	Input
dqs_in0	Data strobe byte 0 (D[7:0])	0	Input
dqs_out_en_x	Output buffer_enable for DQS pads. (bit per DQS line)	0	Output
AXI protocol signals			
ARADDR[31:0]	Read address	0	Input
ARBURST[1:0]	Read transaction burst type	0	Input
ARID[7:0]	Read address ID	0	Input
ARLEN[2:0]	Read transaction burst length	0	Input
ARSIZE[1:0]	Read transaction data size	0	Input

Table 30-1. ESDRAMC Signal Properties (continued)

Name	Function	Reset State	Direction
AWADDR[31:0]	Write address	0	Input
AWBURST[1:0]	Write transaction burst type	0	Input
AWID[7:0]	Write address ID	0	Input
AWLEN[2:0]	Write transaction burst length	0	Input
AWSIZE[1:0]	Write transaction data size	0	Input
WDATA[63:0]	Write data	0	Input
WSTRB[7:0]	Write data strobe	0	Input
ARVALID	Read address valid	0	Input
AWVALID	Write address valid	0	Input
BREADY	Response channel ready	1	Input
RREADY	Read data channel ready	1	Input
WLAST	Last write data indication	0	Input
WVALID	Write data valid	0	Input
WID[7:0]	Write data ID	0	Input
BID[7:0]	Response ID	0	Output
BRESP[1:0]	Response	0	Output
RDATA[63:0]	Read data	0	Output
RID[7:0]	Read data ID	0	Output
RRESP[1:0]	Read response	0	Output
ARREADY	Read address channel ready	1	Output
AWREADY	Write address channel ready	1	Output
BVALID	Response valid	0	Output
RLAST	Last read data indication	0	Output
RVALID	Read data valid	0	Output
WREADY	Write data channel ready	1	Output
CLK	AXI clock	0	Input
IPS protocol signals			
ips_module_en	Register interface enable	0	Input
ips_wdata[31:0]	Register interface write data	0	Input
ipg_clk	Register interface clock	0	Input
ips_addr[11:2]	Register interface address	0	Input
ips_rwb	Register interface read/write bit	0	Input
ips_rdata[31:0]	Register interface read data	0	Output

Table 30-1. ESDRAMC Signal Properties (continued)

Name	Function	Reset State	Direction
ips_xfr_wait	Register interface wait state bit	0	Output
ips_xfr_err	Register interface error state bit	0	Output
Debug signals			
delay_line_significant_change	A signal to indicate a significant change was made to the delay line measurement	0	Output

30.5 Memory Map and Register Definition

The Enhanced SDRAM Controller programming model consists of control and configuration registers (32-bit length) for each chip select and a miscellaneous register, as shown in Table 30-2. The control register maintain system dependent information, while the configuration register maintain memory device dependent information. ESDCFG0 defines the operating characteristics for the region selected by $\overline{CSD0}$, and ESDCFG1 does the same for the $\overline{CSD1}$ region. Bit field assignments within the registers are identical, so a single description will apply to both registers.

Both the control and configuration registers are 32 bits in length with bit fields defined in Figure 30-4 through Figure 30-7 respectively. All implemented bits are fully readable and writable. Reserved bit locations are unaffected by writes and always read back as zero. All register accesses must be SINGLE word (32-bit) operations through the IPS bus protocol. Accesses of any other size will have indeterminate results.

The reset state of each bit is shown underneath the bit field name. An asterisk indicates that the value is dependent on the operating mode selected during reset. Details are provided in the following bit field descriptions.

30.6 Memory Map

The ESDRAMC supports 64, 128, 256, 512 -Mb, 1 -Gb and 2 -Gb, 4 bank,8 bank, single and double data rate, synchronous DRAMs on two independent chip selects. Each chip selects defines a specific memory address mapped as shown in Table 30-3.

Table 30-2 shows the ESDRAMC memory map and Table 30-5 shows the ESDRAMC register definition.

Table 30-2. ESDRAMC Memory Map

Address	Register	Access	Reset	Section/Page
0x83FD_9000 (ESDCTL0)	Enhanced SDRAM Control Register 0	R/W	0x0111_0000	30.7.1.1/30-15
0x83FD_9004 (ESDCFG0)	Enhanced SDRAM Configuration Register 0	R/W	0xE996_68BA	30.7.1.2/30-18
0x83FD_9008 (ESDCTL1)	Enhanced SDRAM Control Register 1	R/W	0x0112_0000	30.7.1.1/30-15
0x83FD_900C (ESDCFG1)	Enhanced SDRAM Configuration Register 1	R/W	0xE99A_64A7	30.7.1.2/30-18

Table 30-2. ESDRAMC Memory Map (continued)

0x83FD_9010 (ESDMISC)	Enhanced SDRAM Miscellaneous Register	R/W	0x000A_4680	30.7.1.3/30-31
0x83FD_9014 (ESDSCR)	Enhanced SDRAM Special Command Register	R/W	0x0000_8000	30.7.1.4/30-34
0x83FD_9020 (ESDCDLY1)	Enhanced SDRAM Delay Line 1 Configuration Debug Register	R/W	0x00F4_8000	30.7.1.6/30-37
0x83FD_9024 (ESDCDLY2)	Enhanced SDRAM Delay Line 2 Configuration Debug Register	R/W	0x00F4_8000	30.7.1.7/30-38
0x83FD_9028 (ESDCDLY3)	Enhanced SDRAM Delay Line 3 Configuration Debug Register	R/W	0x00F4_8000	30.7.1.8/30-39
0x83FD_902C (ESDCDLY4)	Enhanced SDRAM Delay Line 4 Configuration Debug Register	R/W	0x00F4_8000	30.7.1.9/30-40
0x83FD_9030 (ESDCDLY5)	Enhanced SDRAM Delay Line 5 Configuration Debug Register	R/W	0x00F4_8000	30.7.1.10/30-42
0x83FD_9034 (ESDGPR)	Enhanced SDRAM General Purpose Register	R/W	0x2002_00xx	30.7.1.11/30-43
0x83FD_9038 (ESDPRCT0)	Enhanced SDRAM Precharge Counter CS0	R/W	0x0000_0000	30.7.1.12/30-45
0x83FD_903C (ESDPRCT1)	Enhanced SDRAM Precharge Counter CS1	R/W	0x0000_0000	30.7.1.12/30-45

Table 30-3. Chip selects memory region

Address	Use	Access
0x{base_via,000_0000}–0x{base_via,FFF_FFFF}	CSD0 memory region (256MB)	Read/Write
0x{base_bia+1,000_0000}–0x{base_via+1,FFF_FFFF}	CSD1 memory region (256MB)	Read/Write

30.7 Register Summary

Figure 30-2 shows the key to the register fields and Table 30-4 shows the register figure conventions.

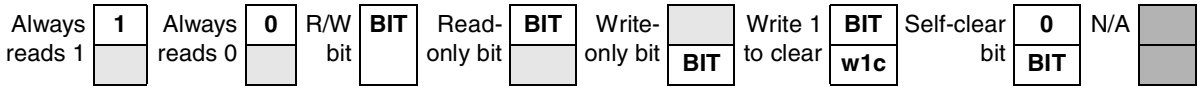


Figure 30-2. Key to Register Fields

Table 30-4. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read-only. Writing this bit has no effect.

Table 30-4. Register Figure Conventions (continued)

Convention	Description
W	Write-only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 30-5 shows the ESDRAMC register summary.

Table 30-5. ESDRAMC Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x83FD_9000 (ESDCTL0)	R	SDE	SREFR			0	ROW			DBL_tRFC	0	COL		0	0	DSIZ	
	W																
	R	SRT		PWDT		0	0	0	0	0	0	0	0	0	0	0	0
	W																
0x83FD_9004 (ESDCFG0)	R	tRFC				tXSR				tXP			tWTR	tRP		tMRD	
	W																
	R	tRAS				tRRD		0	0	tWR	tRCD			tRC			
	W																
0x83FD_9008 (ESDCTL1)	R	SDE	SREFR			0	ROW			DBL_tRFC	0	COL		0	0	DSIZ	
	W																
	R	SRT		PWDT		0	0	0	0	0	0	0	0	0	0	0	0
	W																
0x83FD_900C (ESDCFG1)	R	tRFC				tXSR				tXP			tWTR	tRP		tMRD	
	W																
	R	tRAS				tRRD		0	0	tWR	tRCD			tRC			
	W																

Table 30-5. ESDRAMC Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x83FD_9010 (ESDMISC)	R	CS0_RDY	CS1_RDY	ODT_IDLE_ON	SD-CLK EXT	TERM_CTL3		TERM_CTL2		TERM_CTL1		TERM_CTL0		AP bit			
	W																
	R	DIFF_DQS_EN	AUTO_DLL_PAUSE	ODT_EN	AI	FRC_MSR	MIF3_MODE		RALAT		DDR2_8_BANK	LHD	DDR2_EN	DDR_EN	0	RST	0
	W																
0x83FD_9014 (ESDSCR)	R	0	PSEUDO_ADDR														
	W																
	R	CON_REQ	CON_ACK	0	0	0	0	0	0	0	MAN_DLL_PAUSE	CMD			CS	BA	
	W																
0x83FD_9018 (ESDDAR)	R	DEBUG_ADDR															
	W																
	R	DEBUG_ADDR															
	W																
0x83FD_9020 (ESDCDLY1)	R	SEL_DLY_REG_1	0	0	0	0	0	0	0	DLY_OFFSET_1							
	W																
	R	DLY_ABS_OFFSET_1								DLY_REG_1							
	W																
0x83FD_9024 (ESDCDLY2)	R	SEL_DLY_REG_2	0	0	0	0	0	0	0	DLY_OFFSET_2							
	W																
	R	DLY_ABS_OFFSET_2								DLY_REG_2							
	W																

Table 30-5. ESDRAMC Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x83FD_9028 (ESDCDLY3)	R	SEL_DLY_REG_3	0	0	0	0	0	0	0	DLY_OFFSET_3							
	W																
	R	DLY_ABS_OFFSET_3								DLY_REG_3							
	W																
0x83FD_902C (ESDCDLY4)	R	SEL_DLY_REG_4	0	0	0	0	0	0	0	DLY_OFFSET_4							
	W																
	R	DLY_ABS_OFFSET_4								DLY_REG_4							
	W																
0x83FD_9030 (ESDCDLY5)	R	SEL_DLY_REG_5	0	0	0	0	0	0	0	DLY_OFFSET_5							
	W																
	R	DLY_ABS_OFFSET_5								DLY_REG_5							
	W																
0x83FD_9034 (ESDGPR)	R	DIG_EN	DIG_CYC		DIG_QTR		DIG_OFF0		DIG_OFF1		DIG_OFF2		DIG_OFF3		0	SCT	
	W																
	R	0	0	0	0	0	0	0	0	QTR_CYCLE_LENGTH							
	W																
0xBASE_1038 (ESDPRCT0)	R	0	0	0	0	0	0	0	0	PRCT7			PRCT6			PRCT5	
	W																
	R	PRCT5	PRCT4			PRCT3			PRCT2			PRCT1			PRCT0		
	W																
0xBASE_103C (ESDPRCT1)	R	0	0	0	0	0	0	0	0	PRCT7			PRCT6			PRCT5	
	W																
	R	PRCT5	PRCT4			PRCT3			PRCT2			PRCT1			PRCT0		
	W																

30.7.1 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of the register bit and field function follow the register diagrams, in bit order.

NOTE

Memory may be viewed from either a big-endian or little-endian byte ordering perspective depending on the processor configuration (see Figure 30-3). In big-endian mode (the typical default operating mode), the most significant byte (byte 0) of word 0 is located at address 0. For little-endian mode, the most significant byte of word 0 is located at address 3. Within registers, bits are numbered within a word starting with bit 31 as the most significant bit. By convention, byte 0 of a register is the most significant byte regardless of endian mode.

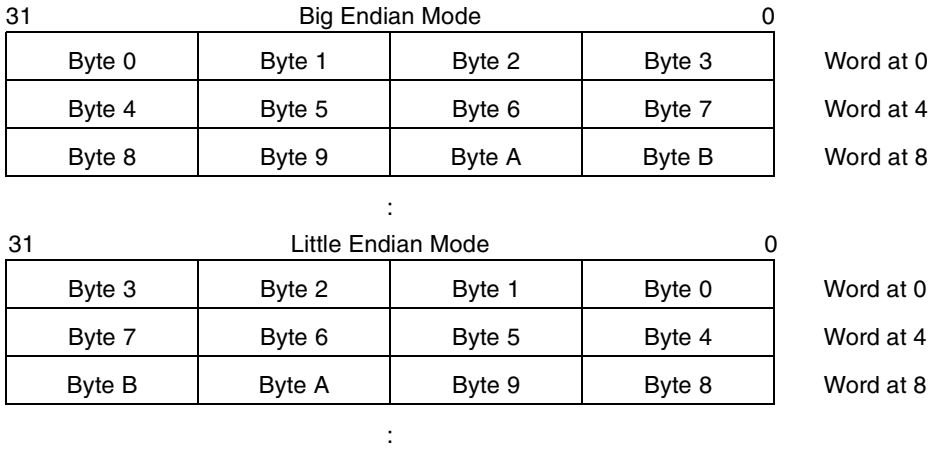


Figure 30-3. Data Organization in Memory

30.7.1.1 ESDCTL0 and ESDCTL1 Control Registers

This register contains the controlling various memory and control settings for the ESDRAMC. The bit assignments for the register are shown in Figure 30-4 and Figure 30-5. The field descriptions for the bit assignments are listed in Table 30-6.

Address 0x83FD_9000 (ESDCTL0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SDE_0	SREFR_0			0	ROW_0			DBL_tRFC_0	0	COL_0		0	0	DSIZ_0	
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRT_0		PWDT_0		0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-4. Enhanced SDRAM Control Register (ESDCTL0)

Address 0x83FD_9008 (ESDCTL1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SDE_1	SREFR_1			0	ROW_1			DBL_tRFC_1	0	COL_1		0	0	DSIZ_1	
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRT_1		PWDT_1		0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-5. Enhanced SDRAM Control Register (ESDCTL1)

Table 30-6. Enhanced SDRAM Control Register (ESDCTL0/1) Field Descriptions

Field	Description
31 SDE	Enhanced SDRAM Controller Enable This control bit enables/disables the Enhanced SDRAM controller. The reset value of this bit is “0”. No clocks and clock enable will be sent to memory. Writing one to those control bits enables the module, for both chip selects. Init sequence of 400 µs will pass from the enable time until the memory will be accessible. Clearing those bits disables the module. By disabling both bits (SDE0 and SDE1) all clocks within the module shuts off (with the exception of register accesses). 0 Disabled 1 Enabled
30–28 SREFR	SDRAM Refresh Rate This control bit field enables/disables SDRAM refresh cycles and controls the refresh rate. Refresh cycles are referenced to a 32 kHz clock. At each falling edge 1, 2, 4, 8 or 16 rows are refreshed as determined by this bit field. Multiple refresh cycles will be separated by the row cycle delay specified in the SRC control field. Refresh is disabled by hardware reset. SREFR Bit Field Encoding settings are listed in Table 30-7.
27	Reserved
26–24 ROW	Row Address Width This control field specifies the number of row addresses used by the memory array. It will affect the way an incoming address will be decoded. 000 11 Row Addresses 001 12 Row Addresses 010 13 Row Addresses 011 14 Row Addresses 100 through 111 are reserved
23 DBL_tRFC	Double tRFC value that was chosen in bits [31–28] of ESDCFG register. 0 tRFC value that will be used will be as shown in tRFC field description. 1 tRFC value that will be used will be double than the value that is shown in tRFC field description.
22	Reserved
21–20 COL	Column Address Width The COL control field is used to specify the number of column addresses in the memory array. It will determine how an incoming address will be decoded. 00 8 bits 01 9 bits 10 10 bits 11 Reserved
19–18	Reserved
17–16 DSIZ	SDRAM Memory Data Width This field defines the width of the SDRAM memory and its alignment on the external data bus. 16-bit ports may be aligned to either the high or low half word to equalize capacitive loading on the bus. Data qualifier mask control outputs must be matched to the selected data bus alignment. Memories aligned to D[31:16] use DQM2 and DQM3. Memories aligned to D[15:0] use DQM0 and DQM1. 00 16-bit memory width aligned to D[31:16] 01 16-bit memory width aligned to D[15:0] (reset value for CSD0) 10 32-bit memory width (reset value for CSD1) 11 reserved

Table 30-6. Enhanced SDRAM Control Register (ESDCTL0/1) Field Descriptions (continued)

Field	Description
15–14 SRT	Self Refresh Timer This field determines whether the SDRAM will be placed in a Self Refresh condition after a selectable delay from the last access. The Self Refresh time-out can be triggered on either the absence of an active bank (SRT=01) or a clock (ESDCTL_CLK) count from the last access (SRT=10 or 11). Count based time-outs do not force the SDRAM into an idle condition (for example, any active banks remain open). The Self Refresh timers feature is disabled by hardware reset. A listing of the SRT Bit Field Encoding is shown in Table 30-9.
13–12 PWDT	Power Down Timer This field determines whether the SDRAM will be placed in a Power Down condition after a selectable delay from the last access. The Power Down time-out can be triggered on either the absence of an active bank (PWDT=01) or a clock (ESDCTL_CLK) count from the last access (PWDT=10 or 11). Count based time-outs performed Precharge all to the SDRAM. The Power Down timer feature is disabled by hardware reset. See sections Section 30.8.3.3, Precharge Power-Down Mode" and Section 30.8.3.4, Active Power-Down Mode" for a comprehensive description of this operating mode. A listing of the PWDT Bit Field Encoding is shown in Table 30-8.
11-0	Reserved

Table 30-7. SREFR Bit Field Encoding

SREFR[2:0]	Rows Each Refresh Clock	# Rows / 64 mS at 32 kHz	Row Rate at 32 kHz
000	Refresh Disabled (bit field reset value)		
001	1	2048	31.25 µs
010	2	4096	15.62 µs
011	4	8192	7.81 µs
100	8	16384	3.91µs
101	16	32768	1.95 µs
110	Reserved		
111	Reserved		

Table 30-8. PWDT Bit Field Encoding

PWDT[1:0]	Power Down Time-out	Memory Device Operating Mode
00	Disabled (bit field reset value)	Run Mode
01	Any time no banks are active [2]	Precharge Power Down
10	64 clocks (ESDCTL_CLK) after completion of last access[1]	Active Power Down
11	128 clocks (ESDCTL_CLK) after completion of last access[1]	Active Power Down

[1] This setting can't be used if the PRCT (precharge timer) or SRT (Self Refresh timer) is enabled.

[2] This is mutually exclusive with setting 01 of SRT field

NOTE

When PWDT is enabled and **lpmd** or **dvfs** requests are required by system, PWDT must be disabled before the requests are granted. PWDT can only be re-enabled after these modes have been exited.

Table 30-9. SRT Bit Field Encoding

SRT[1:0]	Self Refresh Time-out	Memory Device Operating Mode
00	Disabled (bit field reset value)	Run Mode
01	Any time no banks are active [2]	Self-Refresh
10	256 clocks (ESDCTL_CLK) after completion of last access[1]	Precharge all and then self-refresh
11	512 clocks (ESDCTL_CLK) after completion of last access[1]	Precharge all and then self-refresh

[1]This setting ca no't be used if the PRCT (precharge timer) or PWDT (Power Down Timer) is enabled.
[2] This is mutually exclusive with setting 01 of PWDT field
Note: SRT is not supported in DDR2.

30.7.1.2 ESDRAMC Configuration Registers (ESDCFG0/ESDCFG1)

The bit assignments for the configuration registers are shown in Figure 30-6 and Figure 30-7. The field descriptions for the registers is listed in Table 30-10.

Address 0x83FD_9004 (ESDCFG0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	tRFC_0				tXSR_0				tXP_0			tWTR_0	tRP_0		tMRD_0	
Reset	1	1	1	0	1	0	0	1	1	0	0	1	0	1	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	tRAS_0				tRRD_0		0	0	tWR_0	tRCD_0			tRC_0			
Reset	0	1	1	0	1	0	0	0	1	0	1	1	1	0	1	0

Figure 30-6. Enhanced SDRAM Configuration Register 0 (ESDCFG0)

Address 0x83FD_900C (ESDCFG1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	tRFC_1				tXSR_1				tXP_1			tWTR_1	tRP_1		tMRD_1	
Reset	1	1	1	0	1	0	0	1	1	0	0	1	1	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	tRAS_1				tRRD_1		0	0	tWR_1	tRCD_1			tRC_1			
Reset	0	1	1	0	0	1	0	0	1	0	1	0	0	1	1	1

Figure 30-7. Enhanced SDRAM Configuration Register 1 (ESDCFG1)

Table 30-10. ESDCFG0/ESDCFG1 Field Descriptions

Field	Description
31–28 t_{RFC}	Auto refresh to any command. This is the idle delay after auto refresh command until another command can be issued. In DDR2 this is the idle delay after auto refresh command until non read command can be issued. Figure 30-16. 0000 10 clock 0001 11 clock 0010 12 clock 0011 13 clock 0100 14 clock 0101 15 clock 0110 16 clock 0111 17 clock 1000 18 clock 1001 19 clock 1010 20 clock 1011 21 clock 1100 22 clock 1101 23 clock 1110 24 clock 1111 25 clock **Note:** If bit "*DBL_tRFC*" in register "*ESDCTL*" is high, that value that will be chosen in this field will cause double delay. For example, setting tRFC = 0xA and DBL_tRFC = 1 will cause 40 cycles delay.
27–24 t_{XSR}	Exit self refresh to any command. This control field determines the minimum delay between a valid command is issued to the SDRAM after exiting sel-refresh mode. The value programmed in t_{XSR} is the number of clocks inserted after exiting self-refresh mode and any subsequent new valid command. An example timing diagram for t_{XSR} can be found in Figure 30-17. 0000 25 clock 0001 26 clock. 0010 27 clock 0011 28 clock 0100 29 clock 0101 30 clock 0110 31 clock 0111 32 clock 1000 33 clock 1001 34 clock 1010 35 clock 1011 36 clock 1100 37 clock 1101 38 clock 1110 39 clock 1111 40 clock

Table 30-10. ESDCFG0/ESDCFG1 Field Descriptions (continued)

Field	Description
23–21 t_{XP}	Exit power down to next valid command delay This control field determines the minimum delay between a valid command is issued to the memory device after exiting power down mode. The value programmed in t_{XP} is the number of clocks inserted after exiting power down mode and any subsequent new valid command. An example timing diagram for t_{XP} can be found in Figure 30-15. 000 1 cycle delay before new COMMAND issued to LPDDR after power down mode exit 001 2 cycle delay before new COMMAND issued to LPDDR after power down mode exit 010 3 cycle delay before new COMMAND issued to LPDDR after power down mode exit 011 4 cycle delay before new COMMAND issued to LPDDR after power down mode exit 100 5 cycle delay before new COMMAND issued to LPDDR after power down mode exit 101 6 cycle delay before new COMMAND issued to LPDDR after power down mode exit 110 7 cycle delay before new COMMAND issued to LPDDR after power down mode exit 111 8 cycle delay before new COMMAND issued to LPDDR after power down mode exit
20 t_{WTR}	WRITE to READ Command Delay Data for any WRITE burst may be followed by a subsequent READ command. To follow a WRITE without truncating the WRITE burst, t_{WTR} should be set as shown in Figure 30-8. The t_{WTR} should be configured according to the memory device being used. 0 1 clock 1 2 clocks
19–18 t_{RP}	Row Precharge Delay This control bit determines the number of idle clocks inserted between a precharge command and the next row activate command to the same bank.See Figure 30-9. 00 2 clock 01 3 clocks 10 4 clocks 11 5 clocks
17–16 t_{MRD}	Load Mode Register command cycle time This control bits determines the minimum number of idle clocks required after a Load-Mode-Register (LMR). See Figure 30-10. 00 1 clock 01 2 clocks 10 3 clocks 11 4 clocks

Table 30-10. ESDCFG0/ESDCFG1 Field Descriptions (continued)

Field	Description
15–12 t_{RAS}	ACTIVE to PRECHARGE Command These control bits determine the minimum number of clocks required between a ACTIVE to PRECHARGE command to the same bank. See Figure 30-12. 0000 1 clock 0001 2 clocks 0010 3 clocks 0011 4 clocks 0100 5 clocks 0101 6 clocks 0110 7 clocks 0111 8 clocks 1000 9 clock 1001 10 clocks 1010 11 clocks 1011 12 clocks 1100 13 clocks 1101 14 clocks 1110 15 clocks 1111 16 clocks
11–10 t_{RRD}	ACTIVE Bank A to ACTIVE Bank B Command A subsequent ACTIVE command to a different row in the same bank can only be issued after the previous active row has been “closed” (precharged). A subsequent ACTIVE command to another bank can be issued while the first bank is being accessed, which results in a reduction of total row-access overhead. The minimum interval between successive ACTIVE commands to different banks is defined by t_{RRD} as shown in Figure 30-13 (for t_{RRD}=3). The t_{RRD} bits field encoding is listed below and it determines the number of idle clocks inserted between consecutive ACTIVE commands to different banks. 00 1 clock active to active (different banks) 01 2 clocks active to active (different banks) 10 3 clocks active to active (different banks) 11 4 clocks active to active (different banks)
9–8	Reserved.
7 t_{WR}	WRITE to PRECHARGE Command Data for a fixed length WRITE burst may be followed by, or truncated with, a PRECHARGE command to the same bank (provided that auto precharge was not activated). The PRECHARGE command should be issued t_{WR} after the clock edge at which the last desired input data element is registered. t_{WR} control bit determines the number of idle clocks inserted between the last desired input data element and the next PRECHARGE command, as shown in Figure 30-11. 0 2 clocks 1 3 clocks

Table 30-10. ESDCFG0/ESDCFG1 Field Descriptions (continued)

Field	Description
6–4 t_{RCD}	SDRAM Row to Column Delay This field determines the number of clocks inserted between a row activate command and a subsequent read or write command to the same bank.See Figure 30-14. 000 1 clock row to column delay 001 2 clocks row to column delay 010 3 clocks row to column delay 011 4 clocks row to column delay 100 5 clocks row to column delay 101 6 clocks row to column delay 110 7 clocks row to column delay 111 8 clocks row to column delay
3–0 t_{RC}	ACTIVE to ACTIVE, same bank, command delay. 0000 20 clock 0001 2 clocks 0010 3 clocks 0011 4 clocks 0100 5 clocks 0101 6 clocks 0110 7 clocks 0111 8 clocks 1000 9 clocks 1001 10 clocks 1010 11 clocks 1011 12 clocks 1100 13 clocks 1101 14 clocks 1110 15 clocks 1111 16 clocks

Table 30-11 lists and summarizes the Enhanced SDRAM Controller configurable set of timing parameters.

Table 30-11. Configurable Timing Parameters

Symbol	Description	Relevancy
t_{MRD}	Load Mode Register command to ACTIVE or REFRESH command	always
t_{WR}	Write recovery time (write to precharge)	commands to same bank
t_{RAS}	ACTIVE to PRECHARGE command	commands to same bank
t_{RRD}	ACTIVE bank A to ACTIVE bank B command	commands to different banks
t_{RP}	PRECHARGE command period	commands to same bank
t_{RCD}	ACTIVE to READ or WRITE delay	commands to same bank
t_{RC}	ACTIVE to ACTIVE command period	commands to same bank
t_{WTR}	LPDDR READ to WRITE command delay	command to same bank
t_{XP}	LPDDR EXIT power down to next valid command delay	always

Table 30-11. Configurable Timing Parameters (continued)

Symbol	Description	Relevancy
t_{RFC}	Auto refresh to ACTIVE or to another auto refresh command interval.	always
t_{XSR}	SDRAM EXIT self refresh to next valid command delay	always

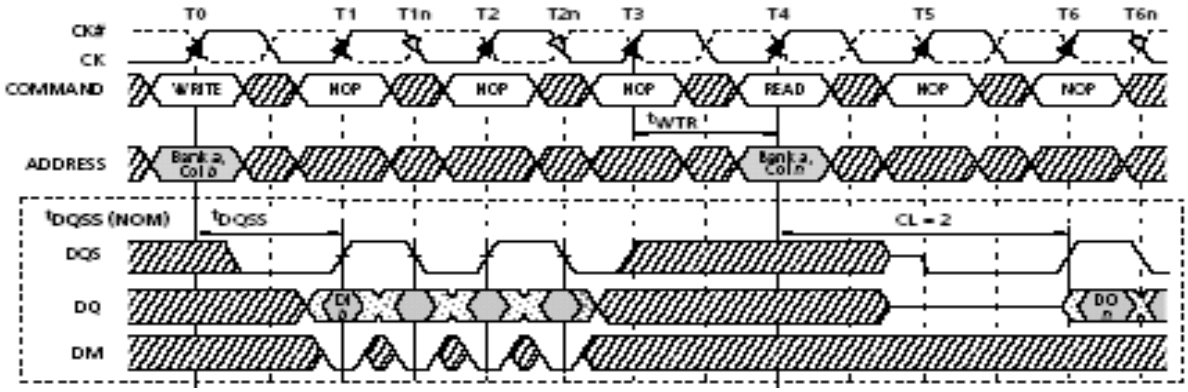


Figure 30-8. tWTR Timing

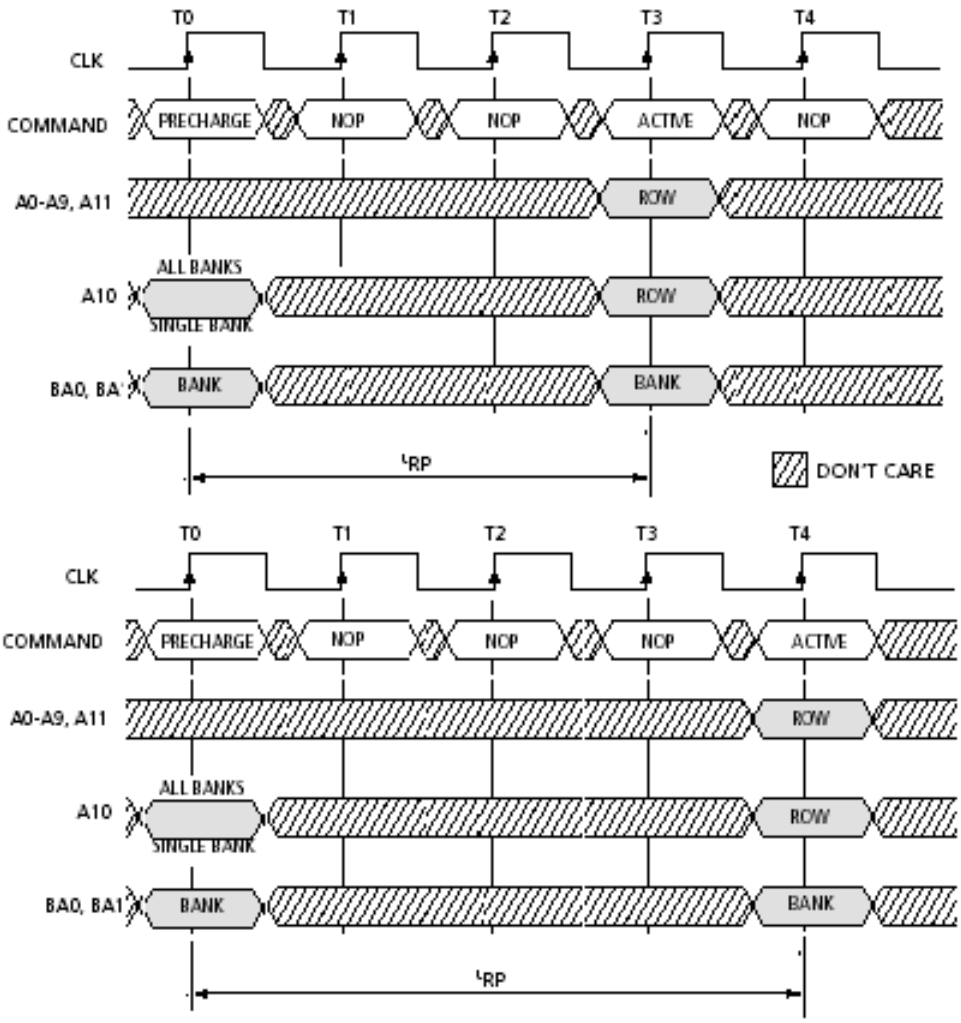


Figure 30-9. tRP—Precharge Delay Timing

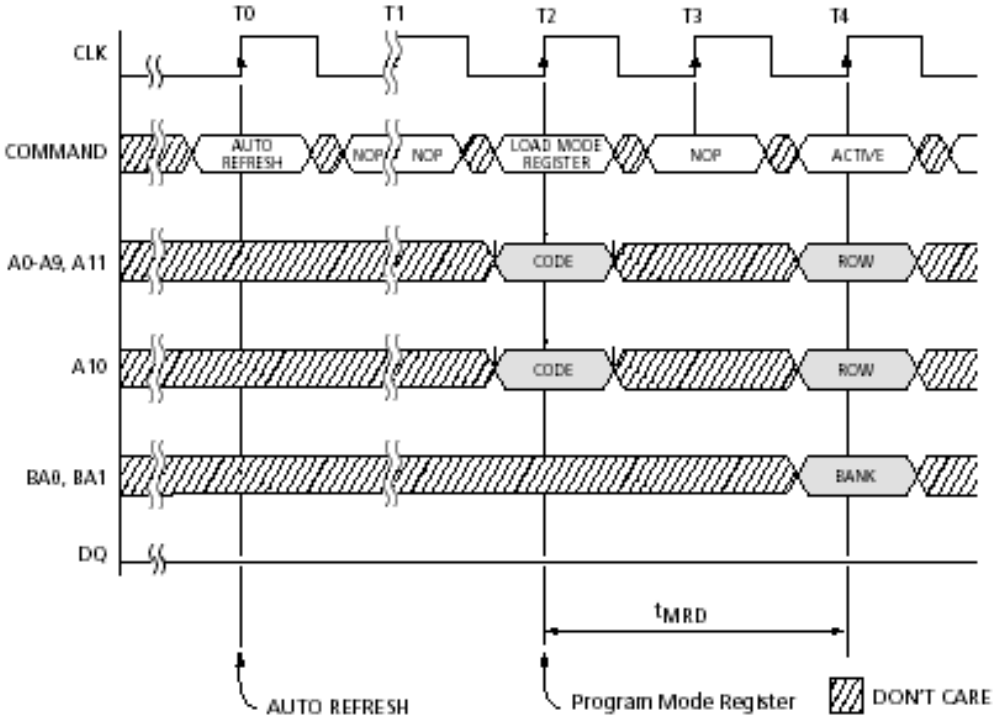


Figure 30-10. tMRD—Load Mode Register cycle time Timing Diagram

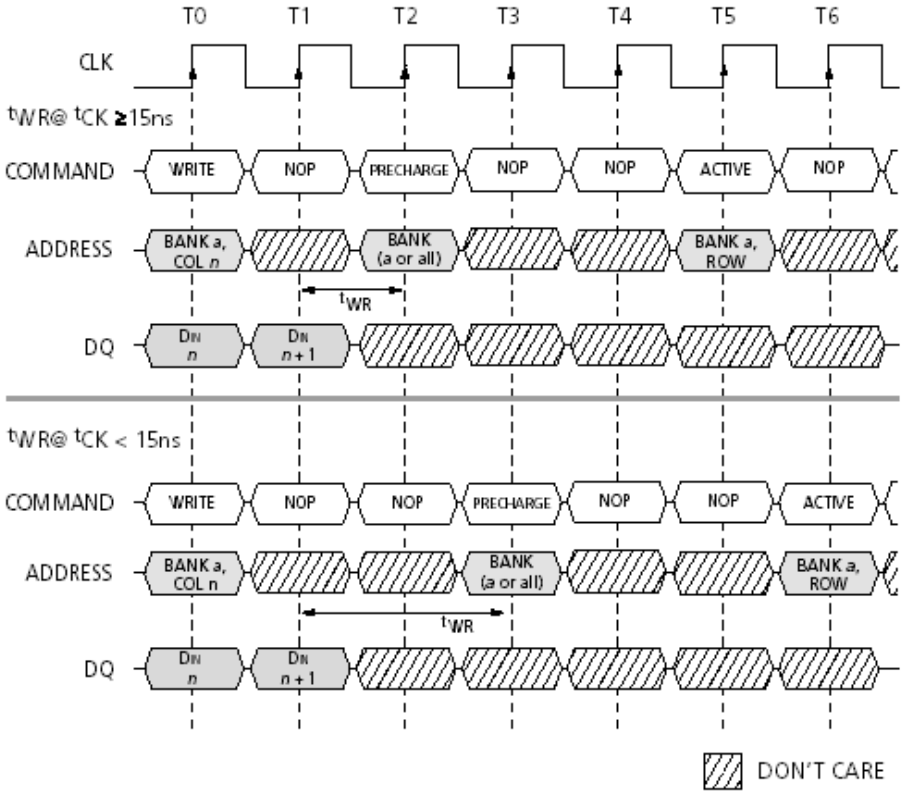


Figure 30-11. tWR—WRITE to PRECHARGE Timing Diagram

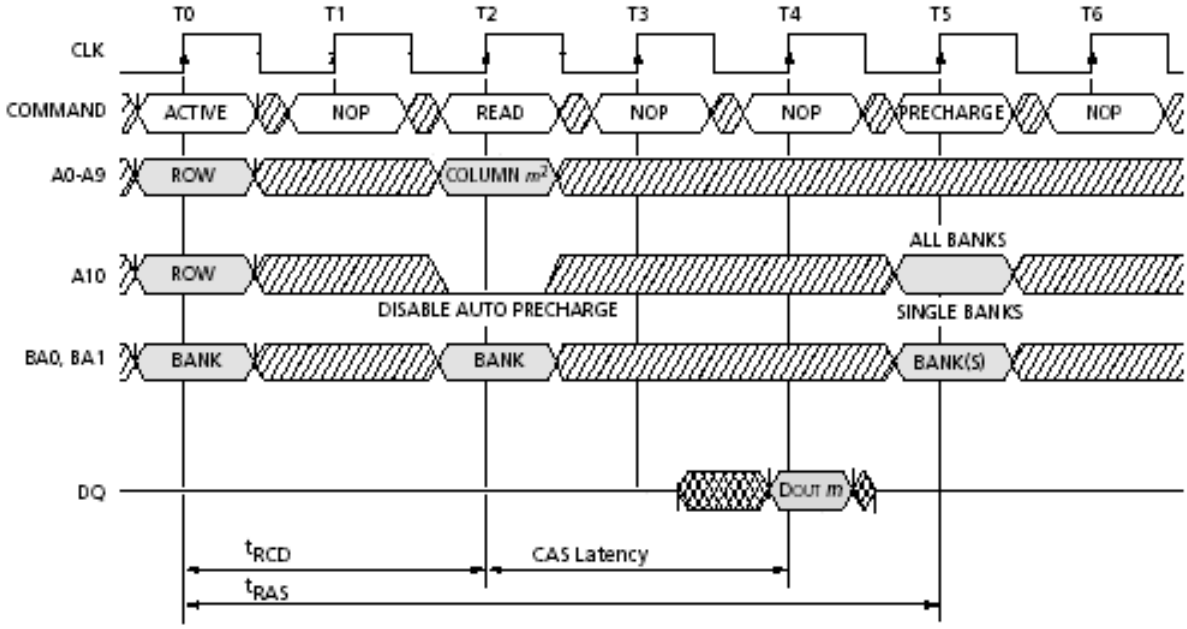


Figure 30-12. t_{RAS}—SDRAM ACTIVE to PRECHARGE Command Timing Diagram

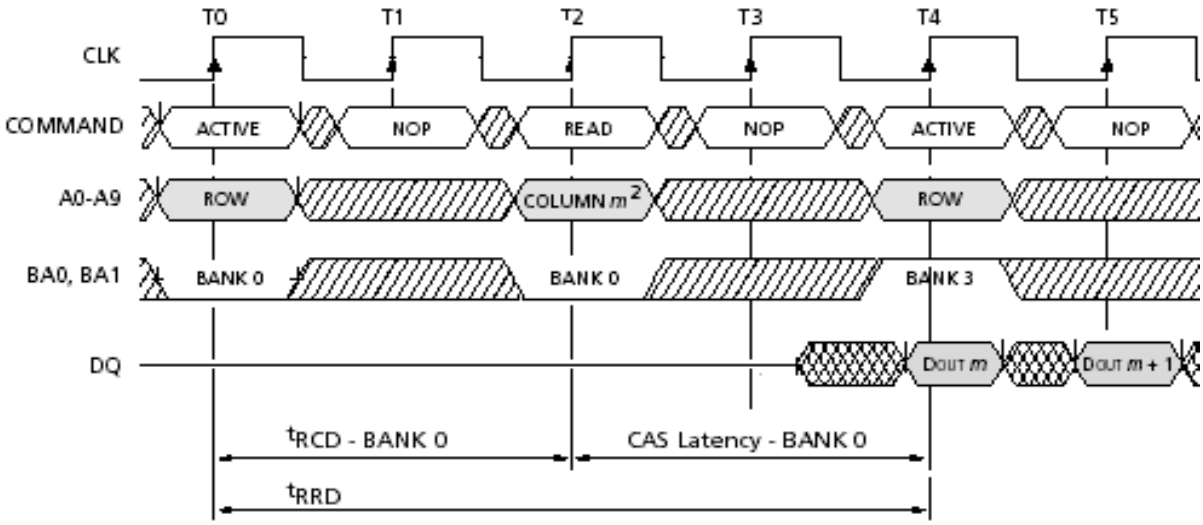


Figure 30-13. t_{RRD}—Alternating Bank Read Access

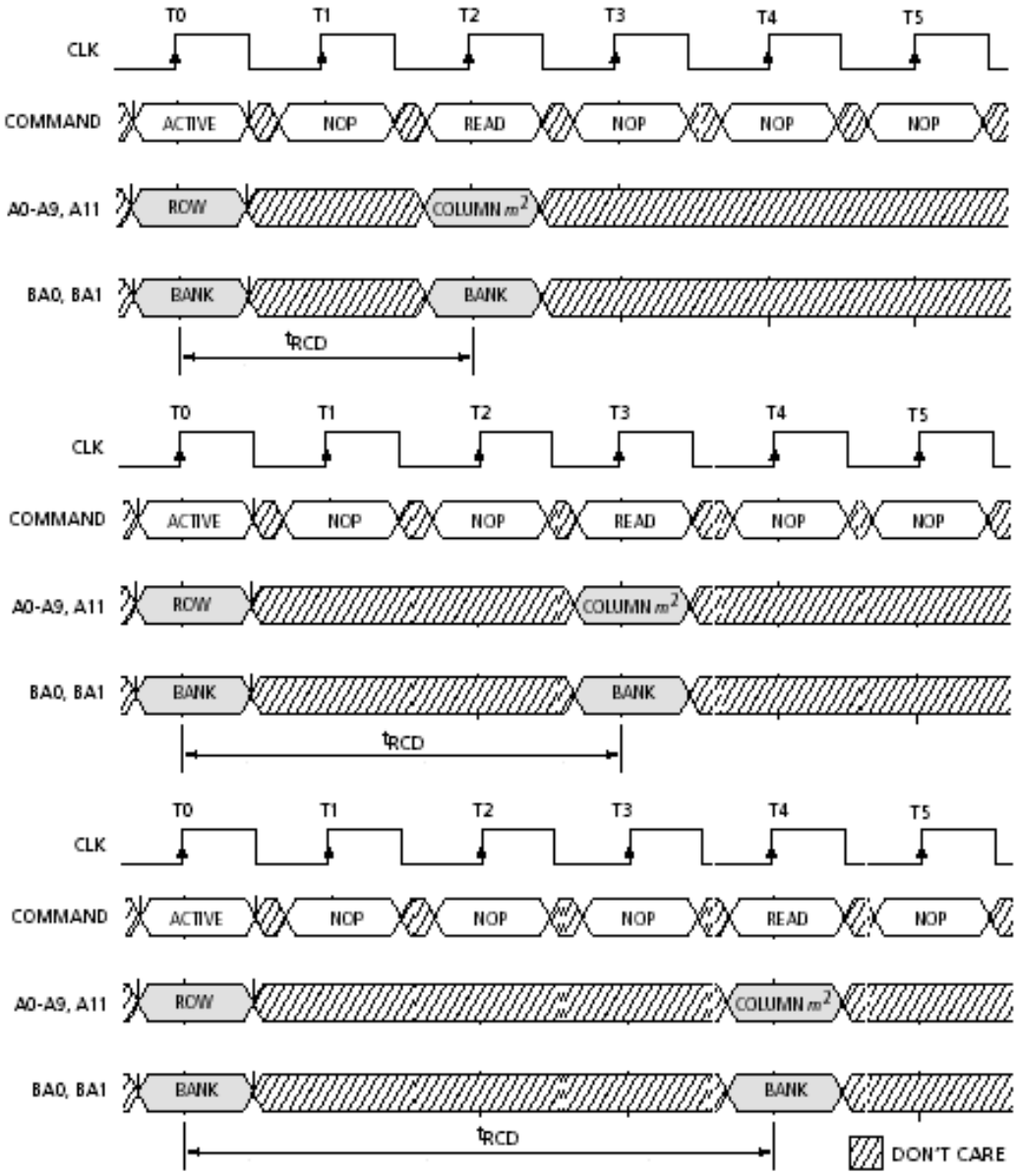


Figure 30-14. t_{RCD}—Row to Column Delay Timing

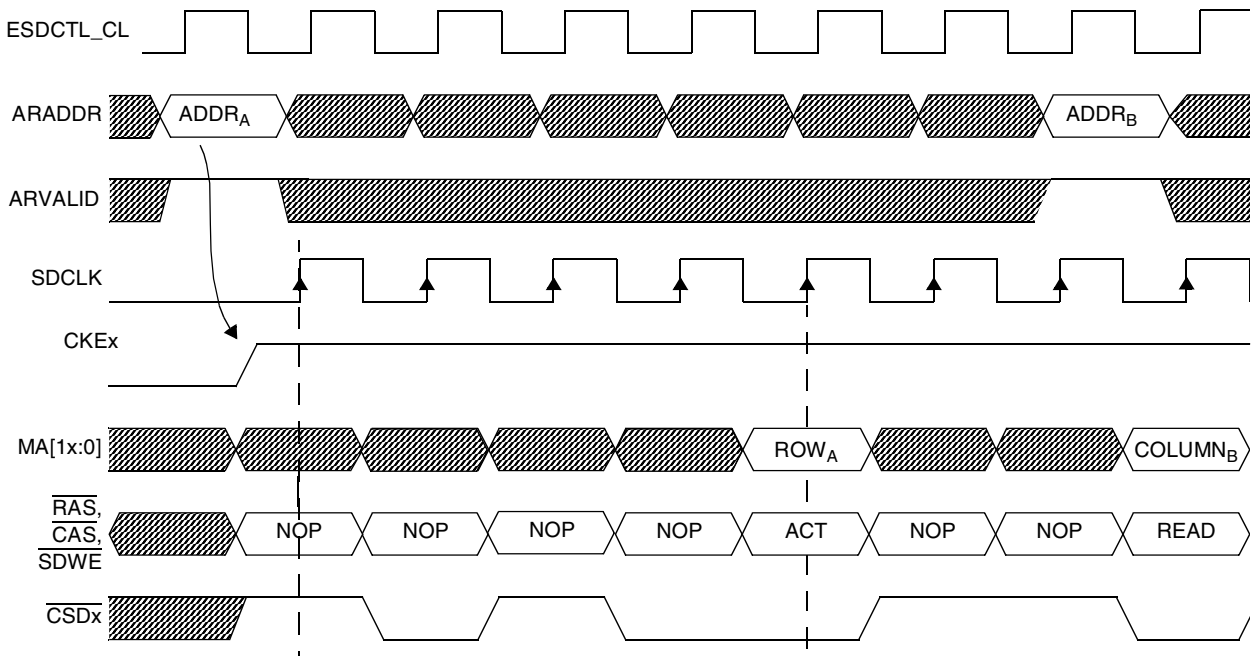


Figure 30-15. t_{XP}—New Command After Power Down Exit (4 cycles)

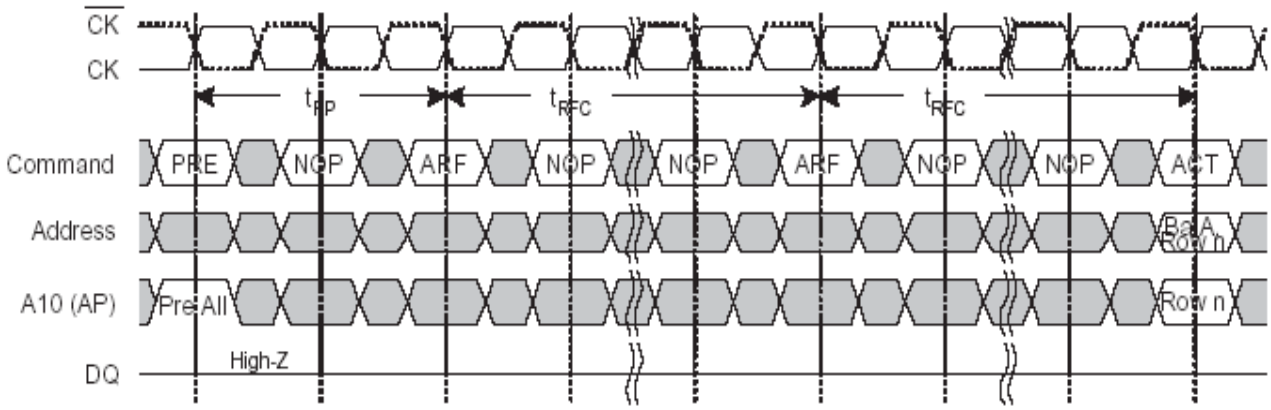


Figure 30-16. t_{RFC}—New Command After Auto Refresh command

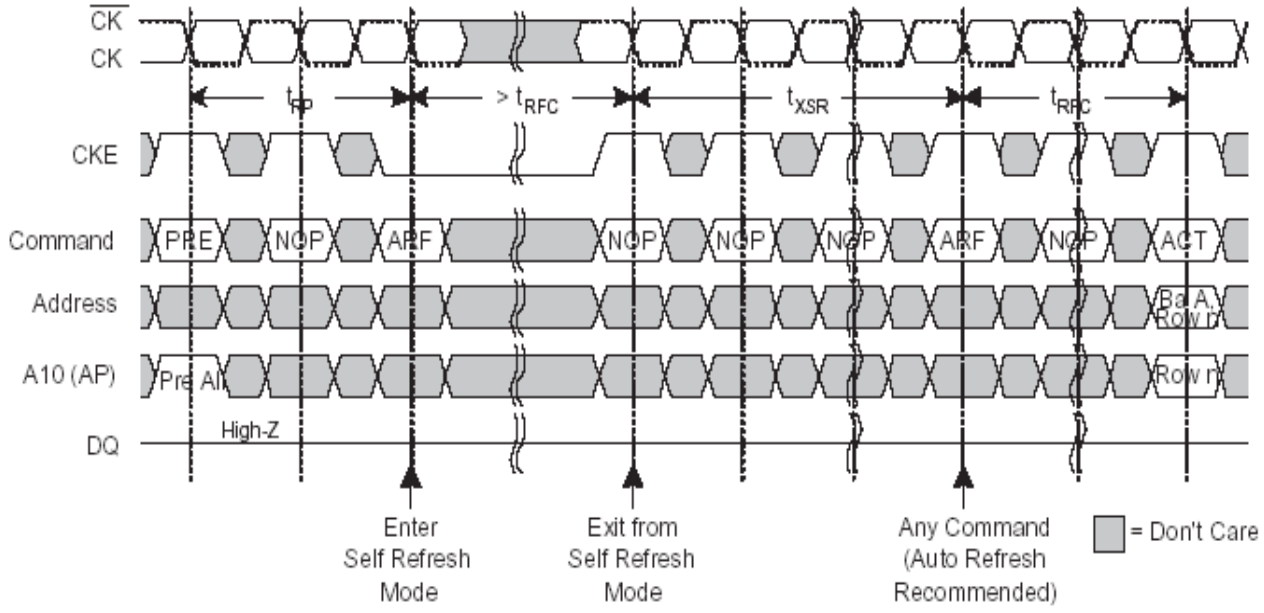


Figure 30-17. t_{RFC}—Refresh Command Period and t_{XSR}—New Command After Self-Refresh Exit

30.7.1.3 ESDMISC Miscellaneous Register (ESDMISC)

This register contains the controlling various memory and control settings for the ESDRAMC. The bit assignments for the register are shown in Figure 30-18, and the field descriptions for the bit assignments are listed in Table 30-12.

Address: 0x83FD_9010 (ESDMISC) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CS0_RDY	CS1_RDY	ODT_IDLE_ON	SD-CLK EXT	TERM_CTL3		TERM_CTL2		TERM_CTL1		TERM_CTL0		AP bit			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIFF_DQS_EN	AUTO_DLL_PAUSE	ODT_EN	BI_ON	FRC_MSR	MIF3_MODE		RALAT		DDR2_8_BANK	LHD	DDR2_EN	DDR_EN	0	RST	0
W																
Reset	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0

Figure 30-18. ESDRAMC Miscellaneous Register (ESDMISC)

Table 30-12. Enhanced SDRAM Miscellaneous Register (ESDCTMISC) Field Descriptions

Field	Description
31 CS0_RDY	External status device on CS0. This is a read-only status bit, that indicates the state of the external memory device(s). This bit is cleared at reset or at deep power down entry for DDR. This bit is set after 400 µs of external memory wakeup period. 0 Device in wakeup period. 1 Device is ready for initialization.
30 CS1_RDY	External status device on CS1. This is a read-only status bit, that indicates the state of the external memory device(s). This bit is cleared at reset or at deep power down entry for DDR. This bit is set after 400 µs of external memory wakeup period. 0 Device in wakeup period. 1 Device is ready for initialization.
29 ODT_IDLE_ON	ODT behavior in IDLE mode: 0 Internal and external ODT will keep its last value and will not turn off. 1 internal and external ODT will be turned off when controller is IDLE.
28 SDCLK EXT	SDCLK extension. In cases that requires shutting off SDCLK, setting this bit will keep SDCLK working two extra clocks after CKE gets low. 0 No extension of SDCLK. 1 SDCLK will continue working 2 extra clocks after cke required to get low.
26–27 TERM_CTL3	For DDR2, when working in ODT mode these two bits determine the termination resistor of the DQS[3] and DQ[31–24] pads when making READ operations. 00 No termination 01 150 Ω 10 75 Ω 11 50 Ω
24–25 TERM_CTL2	For DDR2, when working in ODT mode these two bits determine the termination resistor of the DQS[2] and DQ[23–16] pads when making READ operations. 00 No termination 01 150 Ω 10 75 Ω 11 50Ω
22–23 TERM_CTL1	For DDR2, when working in ODT mode these two bits determine the termination resistor of the DQS[1] and DQ[15–8] pads when making READ operations. 00 No termination 01 150 Ω 10 75 Ω 11 50 Ω
21–20 TERM_CTL0	For DDR2, when working in ODT mode these two bits determine the termination resistor of the DQS[0] and DQ[7–0] pads when making READ operations. 00 No termination 01 150 Ω 10 75 Ω 11 50 Ω

Table 30-12. Enhanced SDRAM Miscellaneous Register (ESDCTMISC) Field Descriptions (continued)

Field	Description
19–16 AP bit	Auto Precharge bit location. This is the bit in the device that when driven high it signals a PRECHARGE command is actually a PRECHARGE ALL command and relevant to all banks. JEDEC standard defines this bit to be bit 10 of the address, and this is the default value for this register. If the device is not JEDEC complaint, then this field needs to be set to match the device requirement. 0000 - address bit 0 is the AP bit 0001 - address bit 1 is the AP bit ... 1111 - address bit 15 is the AP bit
15 DIFF_DQS_EN	Enables differential DQS mode when working with DDR2 device: 0 Differential DQS disabled. 1 Differential DQS enabled. **Note:** On initialization, memory has to be configured as well for this mode.
14 AUTO_DLL_PAUSE	Enable auto pause of 200 cycles after issuing DLL reset at startup (For DDR2 devices) until first READ command. The controller will realize a DLL reset use asserted if a LMR command use issued with “BA=0” and A8 high. 0 AUTO_DLL_PAUSE is disabled, in this case a manual pause can be issued when needed via the SCR register. 1 AUTO_DLL_PAUSE in enabled.
13 ODT_EN	For DDR2 devices, enable the use of the ODT control output bits to the memory. 0 The use of ODT is disabled and a constant ‘0’ will be issued in this outputs. 1 The use of ODT is enabled. **Note:** This bit configuration is common to both CS0 and CS1.
12 BI_ON	Bank interleaving bit. This bit indicates whether the system is interleaving the address or not. 0 Banks are not interleaved, and address will be decoded as bank-row-column 1 Banks are interleaved, and address will be decoded as row-bank-column Example: For an external memory with 8 bit column width and 12 bit row, the address 0x80000100 would decode to bank 0 row 1 col 0 for non interleaved system, but to bank 1 row 0 and col 0 for interleaved system. The address 0x80100000 would be directed to bank 1 row 0 col 0 for the non interleaved system, and to bank 0 row 400 and col 0 for the interleaved configuration.
11 FRC_MSR	Force measurement. When this bit is set the measurement unit will start a new measurement over and over again until this bit will be cleared.
10–9 MIF3_MODE	Controlling the MIF3 mechanism. 00 Disable MIF3. 01 Enable treatment for pending access only. 10 Enable treatment for pending access only. 11 Enable treatment for both pending and predicted accesses.
8–7 RALAT	Read Additional Latency, which determines when the contoller will retrieve the data from the ESDCTL internal fifo. Using this field to compensate on board/chip delays even in slow/fast frequencies. 00 The data will be retrieved from the fifo at the earliest step possible. 01 The data will be retrieved from the fifo one clock later than in RALAT==0. 10 The data will be retrieved from the fifo one clock later than in RALAT==1. 11 Reserved

Table 30-12. Enhanced SDRAM Miscellaneous Register (ESDCTMISC) Field Descriptions (continued)

Field	Description
6 DDR2_8_BANK	DDR2 device with 8 bank is in use. This bit is common for both chip selects. 0 The controller works with 4 bank device. 1 The controller works with 8 bank device.
5 LHD	Latency Hiding Disable for read operations. When setting this bit and making a read access, the next read access will be allowed only after the last data of the first read was sent to arbitration. 0 Latency Hiding Enable. 1 Latency Hiding Disable
4 DDR2_EN	Regular (non mobile DDR2) device. This bit is common for both chip selects. 0 DDR2 Disabled. (If DDR_EN is also disabled the controller will be in Mobile DDR mode) 1 DDR2 Enabled. (DDR_EN bit should be disabled for this option)
3 DDR_EN	Regular (non mobile) DDR1 device is connected. This bit is common for both chip selects. 0 DDR1 Disabled.(If DDR2_EN is also disabled the controller will be in Mobile DDR mode) 1 DDR1 Enabled. (DDR2_EN bit should be disabled for this option)
2	Reserved
1 RST	Software Initiated Local Module Reset. This bit generate local module reset to the ESDRAMC. Writing a 1 to RST bit will put the controller in reset. All ESDRAMC registers are not affected by the software reset, in order to keep the REFRESH mechanism active as initially configured, so the data is not violated. A burst terminate command is issued to the memory after the soft reset (to terminate any active bursts, in order to prevent potential contention on the DATA pads). **Note:** After soft reset, a Precharge all command must be issued prior to normal usage of the ESDRAMC. 0 Soft Reset disabled. 1 Controller is held in reset.
0	Reserved

30.7.1.4 SDRAM Special Command Register

This register is used to issue special commands on the external device bus (such as load mode register, manual self refresh, manual precharge etc.). Every write to this register is interpreted as a command, and a read from this register shows the last command executed.

Every write to this register results in one special command, and the IP bus asserts ips_xfr_wait as long as the special command is being carried out, thus introducing wait states for the next IP write.

Address 0x83FD_9014 (ESDSCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	PSEUDO_ADDR														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CON_REQ	CON_ACK	0	0	0	0	0	0	0	MAN_DLL_PAUSE	CMD			CS	BA	
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-19. ESDRAMC Special Command Register (ESDSCR)

Table 30-13. Enhanced SDRAM Special Command Register (ESDSCR) Field Descriptions

Field	Description
31	Reserved
30–16 PSEUDO_ADDR	Pseudo Address. This field is only considered for LMR and precharge commands, and determines the value on the address bus for the load mode register and if its a precharge all or not.
15 CON_REQ	Configuration request. When this bit is set it tells the controller the user would like to configure the IP registers, so in order to prevent a clash with the AXI bus, the controller holds the aready and awready signals at low, and will not accept LPMD request. This bit is high out of reset, meaning the controller is waiting on configuration and initialization of external memory before accepting any AXI accesses. Note that this bit should be held high during subsequent accesses to this register if AXI readies in low state are to be maintained. 1 AXI address readies are being held low 0 AXI address readies are operating normally
14 CON_ACK	Configuration acknowledge. This read only bit indicates that the controller is idle and no AXI accesses are pending execution. It is recommended that the master will wait until both CON_REQ and CON_ACK are 1 before preforming any writes on the IP bus. 1 AXI channels state machines are all idle 0 At least one of the AXI state machines is not idle
13–7	Reserved
6 MAN_DLL_PAUSE	In the case that AUTO_DLL_PAUSE is disabled, there is an option to issue manual dll pause with this bit. This pause of 200 cycles until next READ command will by issued if this bit is high when making LMR command, the delay will be issued to the CS selected in the LMR command. 0 MANUAL DLL PAUSE is disabled. 1 MANUAL DLL PAUSE is enabled.

Table 30-13. Enhanced SDRAM Special Command Register (ESDSCR) Field Descriptions (continued)

Field	Description
5–3 CMD	Command. This field contains the command to be executed: 000 No command / Exit low power mode. this value should be used when writing to this register (for con_req for example) but no command is needed, or when exiting manual self refresh or deep power down (CS will be considered). 001 Precharge (CS, BA and PSEUDO_ADDR fields used. for precharge all use pseudo address with bit 10 high, such as 0x400) 010 Auto-Refresh Command (Only CS) 011 Load Mode Register Command (CS, BA, and PSEUDO_ADDR fields used) 100 Manual Self Refresh (only CS) 101 Deep power down (Only CS) (Relevant for DDR mode only) 110 Reserved 111 Reserved
2 CS	Chip Select. This field determines which chip select the command is directed at.
1–0 BA	Bank Address. This field determines which bank within the chip select the command is directed at. Only used by the Precharge command or to denote a Load mode register as an Extended mode register set.

30.7.1.5 SDRAM Debug Address Register

This debug register holds the full address of the last access to return an error response (DECERR or SLVERR). This is a read-only register.

Address 0x83FD_9018 (ESDDAR) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEBUG_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEBUG_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-20. ESDRAMC Debug Address Register (ESDDAR)

Table 30-14. Enhanced SDRAM Debug Address Register (ESDDAR) Field Descriptions

Field	Description
31–0 DEBUG_ADDR	This field holds the full address of the last access to return an error response (DECERR or SLVERR). this is a read-only field.

30.7.1.6 SDRAM Delay Line 1 Configuration Debug Register

This debug register controls delay line 1 functionality, i.e., DQS[0] delay used during READ cycles. It allows to override/manually set the delay of DQS[0] line, that is used during READ cycles of BYTE[0]. The delay line compensates for process variations, and produces a constant delay regardless of the process, temperature and voltage. The bit assignments for the register are shown in Figure 30-21 and the field descriptions for the bit assignments are listed in Table 30-15.

Address 0x83FD_9020 (ESDCDLY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL_DLY_REG_1	0	0	0	0	0	0	0	DLY_OFFSET_1							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY_ABS_OFFSET_1								DLY_REG_1							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-21. SDRAM Delay Line 1 Configuration Debug Register

Table 30-15. Enhanced SDRAM Delay Line 1 Control Register (ESDCDLY1) Field Descriptions

Field	Description
31 SEL_DLY_REG_1	SEL_DLY_REG_1 bit selects the delay used by delay line 1. It selects between a quarter of a cycle (measured) plus or minus the delay line 1 offset field (DLY_OFFSET_1) and delay line 1 register value (DLY_REG_1). 0 Delay line 1 value is a quarter of a cycle (measured) plus or minus the delay line 1 correction factor field. 1 Bypass mode: Delay line 1 value is the value of delay line 1 register field (skipping the measurement).
30–24	Reserved
23–16 DLY_OFFSET_1	This field is the delay line 1 offset value. The offset value is used only if SEL_DLY_REG_1 is cleared. The offset value is used to compensate process dependent unalignments between dqs signal and the corresponding read data (in number of small delay units). The field represents positive and negative numbers using two's compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.

Table 30-15. Enhanced SDRAM Delay Line 1 Control Register (ESDCDLY1) Field Descriptions (continued)

Field	Description
15–8 DLY_ABS_OFFSET_1	Absolute delay offset. This field decides the specific delay line absolute delay in fraction of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (DLY_ABS_OFFSET ÷ 512) × tCK. So for the default value of 128 we get a quarter cycle delay. In Bypass mode (SEL_DLY_REG = 1), this register has no effect on the delay. **Note:** Not all changes will have effect on the actual delay. If the requested change is smaller then the delay line resolution (1 delay unit, which varies between 20 pSec in best case to 50 in worst case), then no change will occur.
7–0 DLY_REG_1	This field is the delay (in number of buffer units) that will be used by delay line 1, if the SEL_DLY_REG1 bit is set. Since the delay is process, temperature and voltage dependent, for a given value of this filed we get different delay values. The SoC team should provide the equivalent delay of 1 buffer. This field contains only positive numbers.

30.7.1.7 SDRAM Delay Line 2 Configuration Debug Register

This debug register controls delay line 2 functionality, i.e., DQS[1] delay used during READ cycles. It allows to override/manually set the delay of DQS[1] line, that is used during READ cycles of BYTE[1]. The bit assignments for the register are shown in Figure 30-22 and the field descriptions for the bit assignments are listed in Table 30-16.

Address: 0x83FD_9024 (ESDCDLY2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL_DLY_REG_2	0	0	0	0	0	0	0	DLY_OFFSET_2							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY_ABS_OFFSET_2								DLY_REG_2							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-22. SDRAM Delay Line 2 Configuration Debug Register

Table 30-16. Enhanced SDRAM Delay Line 2 Control Register (ESDCDLY2) Field Descriptions

Field	Description
31 SEL_DLY_REG_2	SEL_DLY_REG_2 bit selects the delay used by delay line 1. It selects between a quarter of a cycle (measured) plus or minus the delay line 1 offset field (DLY_OFFSET_2) and delay line 1 register value (DLY_REG_2). 0 Delay line 2 value is a quarter of a cycle (measured) plus or minus the delay line 2 correction factor field. 1 Bypass mode: Delay line 2 value is the value of delay line 2 register field (skipping the measurement).
30-24	Reserved

Table 30-16. Enhanced SDRAM Delay Line 2 Control Register (ESDCDLY2) Field Descriptions (continued)

Field	Description
23-16 DLY_OFFSET_2	This field is the delay line 2 offset value. The offset value is used only if SEL_DLY_REG_2 is cleared. The offset value is used to compensate process dependent unalignments between dqs signal and the corresponding read data (in number of small delay units). The field represents positive and negative numbers using twos compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.
15-8 DLY_ABS_OFFSET_2	Absolute delay offset. This field decides the specific delay line absolute delay in fraction of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (DLY_ABS_OFFSET / 512) * tCK. So for the default value of 128 we get a quarter cycle delay. In Bypass mode (SEL_DLY_REG = 1), this register has no effect on the delay. Note that not all changes will have effect on the actual delay. If the requested change is smaller then the delay line resolution (1 delay unit, which varies between 20 pSec in best case to 50 in worst case), then no change will occur.
7-0 DLY_REG_2	This field is the delay (in number of buffer units) that will be used by delay line 1, if the SEL_DLY_REG2 bit is set. Since the delay is process, temperature and voltage dependent, for a given value of this filed we get different delay values. The SoC team should provide the equivalent delay of 1 buffer. This field contains only positive numbers.

30.7.1.8 SDRAM Delay Line 3 Configuration Debug Register

This debug register controls delay line 3 functionality, i.e., DQS[2] delay used during READ cycles. It allows to override/manually set the delay of DQS[2] line, that is used during READ cycles of BYTE[2]. The bit assignments for the register are shown in Figure 30-23 and the field descriptions for the bit assignments are listed in Table 30-17.

Address 0x83FD_9028 (ESDCDLY3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL_DLY_REG_3	0	0	0	0	0	0	0	DLY_OFFSET_3							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY_ABS_OFFSET_3								DLY_REG_3							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-23. SDRAM Delay Line 3 Configuration Debug Register

Table 30-17. Enhanced SDRAM Delay Line 3 Control Register (ESDCDLY3) Field Descriptions

Field	Description
31 SEL_DLY_REG_3	SEL_DLY_REG_3 bit selects the delay used by delay line 3. It selects between a quarter of a cycle (measured) plus or minus the delay line 3 offset field (DLY_OFFSET_3) and delay line 3 register value (DLY_REG_3). 0 Delay line 3 value is a quarter of a cycle (measured) plus or minus the delay line 3 correction factor field. 1 Bypass mode: Delay line 3 value is the value of delay line 3 register field (skipping the measurement).
30–24	Reserved
23–16 DLY_OFFSET_3	This field is the delay line 3 offset value. The offset value is used only if SEL_DLY_REG_3 is cleared. The offset value is used to compensate process dependent unalignments between dqs signal and the corresponding read data (in number of small delay units). The field represents positive and negative numbers using twos compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.
15–8 DLY_ABS_OFFSET_3	Absolute delay offset. This field decides the specific delay line absolute delay in fraction of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (DLY_ABS_OFFSET / 512) * tCK. So for the default value of 128 we get a quarter cycle delay. In Bypass mode (SEL_DLY_REG = 1), this register has no effect on the delay. **Note:** Not all changes affect the actual delay. If the requested change is smaller than the delay line resolution (1 delay unit, which varies between 20 pSec in best case to 50 in worst case), then no change occurs.
7–0 DLY_REG_3	This field is the delay (in number of buffer units) that will be used by delay line 3, if the SEL_DLY_REG3 bit is set. Since the delay is process, temperature and voltage dependent, for a given value of this filed we get different delay values. The SoC team should provide the equivalent delay of 1 buffer. This field contains only positive numbers.

30.7.1.9 SDRAM Delay Line 4 Configuration Debug Register

This debug register controls delay line 4 functionality, i.e., DQS[3] delay used during READ cycles. It allows to override/manually set the delay of DQS[3] line, that is used during READ cycles of BYTE[3].

The bit assignments for the register are shown in Figure 30-24 and the field descriptions for the bit assignments are listed in Table 30-18.

Address 0x83FD_902C (ESDCDLY4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL_DLY_REG_4	0	0	0	0	0	0	0	DLY_OFFSET_4							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY_ABS_OFFSET_4								DLY_REG_4							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-24. SDRAM Delay Line 4 Configuration Debug Register

Table 30-18. Enhanced SDRAM Delay Line 4 Control Register (ESDCDLY4) Field Descriptions

Field	Description
31 SEL_DLY_REG_4	SEL_DLY_REG_4 bit selects the delay used by delay line 4. It selects between a quarter of a cycle (measured) plus or minus the delay line 4 offset field (DLY_OFFSET_4) and delay line 4 register value (DLY_REG_4). 0 Delay line 4 value is a quarter of a cycle (measured) plus or minus the delay line 4 correction factor field. 1 Bypass mode: Delay line 4 value is the value of delay line 4 register field (skipping the measurement).
30–24	Reserved
23–16 DLY_OFFSET_4	This field is the delay line 4 offset value. The offset value is used only if SEL_DLY_REG_4 is cleared. The offset value is used to compensate process dependent unalignments between dqs signal and the corresponding read data (in number of small delay units). The field represents positive and negative numbers using twos compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.
15–8 DLY_ABS_OFFSET_4	Absolute delay offset. This field decides the specific delay line absolute delay in fraction of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (DLY_ABS_OFFSET / 512) * tCK. So for the default value of 128 we get a quarter cycle delay. In Bypass mode (SEL_DLY_REG = 1), this register has no effect on the delay. **Note:** Not all changes affect the actual delay. If the requested change is smaller than the delay line resolution (1 delay unit, which varies between 20 pSec in best case to 50 in worst case), then no change occurs.
7–0 DLY_REG_4	This field is the delay (in number of buffer units) that will be used by delay line 4, if the SEL_DLY_REG4 bit is set. Since the delay is process, temperature and voltage dependent, for a given value of this filed we get different delay values. The SoC team should provide the equivalent delay of 1 buffer. This field contains only positive numbers.

30.7.1.10 SDRAM Delay Line 5 Configuration Debug Register

This debug register controls delay line 5 functionality, i.e., data bus delay used during write cycles. It allows override and manual set for the delay of the data bus during write cycles.

The bit assignments for the register are shown in Figure 30-25 and the field descriptions for the bit assignments are listed in Table 30-19.

Address 0x83FD_9030 (ESDCDLY5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL_DLY_REG_5	0	0	0	0	0	0	0	DLY_OFFSET_5							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLY_ABS_OFFSET_5								DLY_REG_5							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-25. SDRAM Delay Line 5 Configuration Debug Register

Table 30-19. Enhanced SDRAM Delay Line 5 Control Register (ESDCDLY5) Field Descriptions

Field	Description
31 SEL_DLY_REG_5	SEL_DLY_REG_5 bit selects the delay used by delay line 5. It selects between a quarter of a cycle (measured) plus or minus the delay line 5 offset field (DLY_OFFSET_5) and delay line 5 register value (DLY_REG_5). 0 Delay line 5 value is a quarter of a cycle (measured) plus or minus the delay line 5 correction factor field. 1 Bypass mode: Delay line 5 value is the value of delay line 5 register field (skipping the measurement).
30–24	Reserved
23–16 DLY_OFFSET_5	This field is the delay line 5 offset value. The offset value is used only if SEL_DLY_REG_5 is cleared. The offset value is used to compensate process dependent unalignments between dqs signal and the corresponding read data (in number of small delay units). The field represents positive and negative numbers using twos compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.

Table 30-19. Enhanced SDRAM Delay Line 5 Control Register (ESDCDLY5) Field Descriptions (continued)

Field	Description
15–8 DLY_ABS_OFFSET_5	Absolute delay offset. This field decides the specific delay line absolute delay in fraction of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (DLY_ABS_OFFSET / 512) * tCK. So for the default value of 128 we get a quarter cycle delay. In Bypass mode (SEL_DLY_REG = 1), this register has no effect on the delay. **Note:** Not all changes affect the actual delay. If the requested change is smaller than the delay line resolution (1 delay unit, which varies between 20 pSec in best case to 50 in worst case), then no change occurs.
7–0 DLY_REG_5	This field is the delay (in number of buffer units) that will be used by delay line 5, if the SEL_DLY_REG5 bit is set. Since the delay is process, temperature and voltage dependent, for a given value of this filed we get different delay values. The SoC team should provide the equivalent delay of 1 buffer. This field contains only positive numbers.

30.7.1.11 General-Purpose Register

The general-purpose register is a register that shows the number of delay units that fit in a cycle. It is also responsible for the delay_line_significant_change debug signal. The reset value for the QTR_CYCLE field is unknown because the register value is updated from the measured delay after reset. The register value represents the number of delay units required to achieve a delay of 1 clock cycle, as a function of the IC conditions (temperature, voltage, frequency, process).

The other field defines a significant change to the delay measurement. It can be set to 2 delay units (any change which is not a glitch) or a half/quarter/eighth of the measured quarter cycle, to a minimum of two, according to the user's needs.

In addition, it controls the DQS-IN gating logic. This logic is necessary when no pull down/up is placed on the DQS,DQS_B signals. A separate technical note explains how to calibrate these values.

The bit assignments for the register are shown in Figure 30-26 and the field descriptions for the bit assignments are listed in Table 30-20. Bits 31 – 19 are write-only, meaning the user can write to these bits but not cannot read their value.

Address 0x83FD_9034 (ESDGPR) Access: User

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R														0	SCT	
W	DIG_EN	DIG_CYC		DIG_QTR		DIG_OFF0		DIG_OFF1		DIG_OFF2		DIG_OFF3				
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	QTR_CYCLE_LENGTH							
W																
Reset	0	0	0	0	0	0	0	0	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Figure 30-26. General-Purpose Register

Table 30-20. General-Purpose Register (ESDGPR) Field Descriptions

Field	Description
31 DIG_EN	DQS-IN gating en. 0 DQS-IN gating off. 1 DQS-IN gating on - this is needed if no pull up/down is used on DQS/DQS_B bus. When using this option and working with both CS0 and CS1, DSIZ_0 should be equal to DSIZ_1(Registers ESDCTL0 and ESDCTL1 - Memory data size field).
30–29 DIG_CYC	DQS-IN gating location in cycle units. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in 1 cycle. 10 DQS gating will be delayed in 2 cycles. 11 DQS gating will be delayed in 3 cycle.
28–27 DIG_QTR	DQS-IN gating location in 1/4 cycle units. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in an additional 1/4 cycle. 10 DQS gating will be delayed in an additional 2/4 cycle. 11 DQS gating will be delayed in an additional 3/4 cycle.
26–25 DIG_OFF0	DQS-IN gating location offset for DQS[0]. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in an additional 1/4 cycle. 10 DQS gating will be delayed in an additional 2/4 cycle. 11 DQS gating will be delayed in an additional 3/4 cycle. The total delay of DQS[0] gating will equal DIG_CYC + 1/4*DIG_QTR + 1/4*DIG_OFF0(cycles).
24–23 DIG_OFF1	DQS-IN gating location offset for DQS[1]. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in an additional 1/4 cycle. 10 DQS gating will be delayed in an additional 2/4 cycle. 11 DQS gating will be delayed in an additional 3/4 cycle. The total delay of DQS[1] gating will equal DIG_CYC + 1/4*DIG_QTR + 1/4*DIG_OFF1(cycles).

Table 30-20. General-Purpose Register (ESDGPR) Field Descriptions

Field	Description
22–21 DIG_OFF2	DQS-IN gating location offset for DQS[2]. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in an additional 1/4 cycle. 10 DQS gating will be delayed in an additional 2/4 cycle. 11 DQS gating will be delayed in an additional 3/4 cycle. The total delay of DQS[2] gating will equal DIG_CYC + 1/4*DIG_QTR + 1/4*DIG_OFF2(cycles).
20–19 DIG_OFF3	DQS-IN gating location offset for DQS[3]. 00 DQS gating will not be delayed. 01 DQS gating will be delayed in an additional 1/4 cycle. 10 DQS gating will be delayed in an additional 2/4 cycle. 11 DQS gating will be delayed in an additional 3/4 cycle. The total delay of DQS[3] gating will equal DIG_CYC + 1/4*DIG_QTR + 1/4*DIG_OFF3(cycles).
18	Reserved
17–16 SCT	Significant Change Threshold: This field defines a significant change to the delay measurement for the delay_line_significant_change debug signal. This signal toggles high and remains so until the next measurement, if the current measurement was different from the previous one as follows: 00 2 delay units 01 QTR_CYCLE_LENGTH / 2 10 QTR_CYCLE_LENGTH / 4 11 QTR_CYCLE_LENGTH / 8
15–8	Reserved.
7–0 QTR_CYCLE_LENGTH	This value shows the number of delay units that are used for Delay Line 5. It takes into account the absolute delay and the relative delay.

30.7.1.12 ESDPRCT0 and ESDPRCT1 Control Registers

This register contains the controlling various memory and control settings for the ESDRAMC. The bit assignments for the register are shown in Figure 30-27 and Figure 30-28. The field descriptions for the bit assignments are listed in Table 30-21.

Address 0x83FD_9038 (ESDPRCT0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	PRCT7				PRCT6			PRCT5
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRCT5	PRCT4			PRCT3				PRCT2				PRCT1			PRCT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-27. CS0 precharge counter register (ESDPRC0)

Address 0x83FD_903C (ESDPRCT1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	PRCT7			PRCT6			PRCT5	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRCT5	PRCT4			PRCT3			PRCT2			PRCT1			PRCT0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 30-28. CS1 precharge counter register (ESDPRC1)

Table 30-21. Precharge Counter Register Field Description

Field	Description
31–24	Reserved.
23–21 PRCT7	Precharge Timer for bank 7 Precharges bank 7 after 2^PRCT clocks of no activity. Table 30-22 illustrates the PRCT bit field encoding. “Closing” (due to precharge command) the last used/open row in any non active bank within a chip select reduces the power consumption of the external memory device. The power saving is device dependent, and one should consult/examine the external memory device specification for more details on power consumption reduction. If PRCT is enabled, a PRECHARGE command is issued after approximately number of cycles (as shown in Table 30-22) of non activity to one of the SDRAM/LPDDR banks. The number of cycles before the PRECHARGE command is issued depends on command bus (WE, RAS, CAS and CSD) availability (means there is no active access to other bank) and the memory timing parameters.
20–18 PRCT_B6	Same as PRCT7 but for bank 6.
17–15 PRCT_B5	Same as PRCT7 but for bank 5.
14–12 PRCT_B4	Same as PRCT7 but for bank 4.
11–9 PRCT_B3	Same as PRCT7 but for bank 3.
8–6 PRCT_B2	Same as PRCT7 but for bank 2.
5–3 PRCT_B1	Same as PRCT7 but for bank 1.
2–0 PRCT_B0	Same as PRCT7 but for bank 0.

Table 30-22. PRCT Field Encoding

PRCT[2:0][1]	Precharge Timer[2]
000	Disabled (Bit field reset value)
001	2 clocks to precharge
010	4 clocks to precharge
011	8 clocks to precharge
100	16 clocks to precharge
101	32 clocks to precharge
110	64 clocks to precharge
111	128 clocks to precharge

1 The PRCT can be used only if PWDT/SRT is disabled (“00”) or set to “any time no banks are active (“01”)”. PRCT can’t be used with any other PWDT/SRT settings.

2 The number of clocks is approximate and it depends on the external bus availability

30.8 Functional Description

This section addresses General Enhanced SDRAM Controller operating characteristics. The discussion starts with the optimization strategy and continues with the address decoding, one of the most basic of all DRAM controller features. The following subsections explain operation out of reset, hardware refresh, and the low-power modes. Each of the Enhanced SDRAM Controller operating modes are also described.

The Enhanced SDRAM Controller is designed to support a broad range of JEDEC standard DDR2/LPDDR configurations, including devices of 64-, 128-, 256-, 512-Mbyte, 1- and 2-Gbyte densities. Due to the physical size constraints of the target applications, the design support memory devices with data widths of 16 and 32 bits. The controller supports mDDR memory devices of 4 banks and DDR2 memory devices of 4 or 8 banks.The controller supports memory devices of up to 200 MHz, which work in CAS latency of 3.

30.8.1 Enhanced SDRAM Controller Optimization Strategy

The ESDCL receives read and write accesses through its AXI channels. When an access is accepted, the controller lowers its aready and awready signals until the last command for the accepted access is sent to the memory. Then, the aready and awready signals rise again so that they can accept the next access.

The controller can be configured to prepare for next incoming accesses. This configuration increases the chances that the next incoming access will be accepted in hit state. Hit accesses can be treated more quickly than miss accesses because no active or precharge commands are needed.

The controller optimizes through the use of the two sideband signal channels: mif3_p1 and mif3_p2.

- mif3_p1 provides the chip select, bank, and row information of the waiting access on AXI channel.
- mif3_p2 provides the chip select, bank, and row information of the next incoming miss access, based of M4IF prediction mechanism.

The controller uses every NOP cycle in its working mode to prepare the two sideband signal channels. It prepares mif3_p1 first, followed by mif3_p2. This increases the probability that the accepted accesses will be hit accesses.

For maximum performance, set the MIF3_MODE field in the ESDMISC register to 3.

30.8.1.1 Multiplexed Address Bus

Table 30-23 illustrates how a CPU address is scrambled by the Enhanced SDRAM Controller to implement a contiguous address space.

Table 30-23. CPU to memory Translation

CPU ADDRESS	16-bit SDRAM[1]	32-bit SDRAM[1]
A25	—	BA1
A24	BA1	BA0
A23	BA0	R11
A22	R11	R10
A21	R10	R9
A20	R9	R8
A19	R8	R7
A18	R7	R6
A17	R6	R5
A16	R5	R4
A15	R4	R3
A14	R3	R2
A13	R2	R1
A12	R1	R0
A11	R0	C9
A10	C9	C8
A9	C8	C7
A8	C7	C6
A7	C6	C5
A6	C5	C4
A5	C4	C3
A4	C3	C2
A3	C2	C1
A2	C1	C0

Table 30-23. CPU to memory Translation (continued)

CPU ADDRESS	16-bit SDRAM[1]	32-bit SDRAM[1]
A1[2]	C0	-
A0[3]	—	-

1 For this example a memory configuration with 10 columns and 12 rows is illustrated. The address translation is based on the following concept, COLUMN - ROW - BANK.

2 CPU A1 defines how the data masks are driven, i.e., it is used as the byte enable for non word accesses. This bit has a regular/normal use only in case of 16-bit memory, while CPU A0 defines the 2 bytes (low and high) in the 16-bit word.

3 CPU A0 defines how the data masks are driven, i.e., it is used as the byte enable for non word accesses to 32-bit memory device. Both CPU A0 and A1 defines the 4 bytes in the 32-bit word.

Note: C=COLUMN, R=ROW, BA=BANK

The Enhanced SDRAM Controller multiplexed address bus is aligned to the column addresses so that address line A1 always appears on pin MA0 for 16-bit memory devices and A2 always appears on MA0 for 32-bit memory devices. With this alignment, the “folding point” in the multiplexor is driven solely by the number of column address bits. Column bus widths of 8 to 11 bits are supported.

Table 30-24 summarizes the multiplex options supported by the controller for 16 and 32-bit devices respectively. Column addresses through A10 are driven regardless of the multiplexor configuration, although some of the lines will be unused for the smaller page sizes.

Table 30-24. Address Multiplexing by Column/Row Width for 16-bit Devices

Device Pins	ESDCTL Pins	16-bit LPDDR Memory Device									
		64 Mbytes		128 Mbytes		256 Mbytes		512 Mbytes		1 Mbytes	
		8 col 12 row		9 col 12 row		9 col 13 row		10 col 13 row		10 col 14 row	
		Col	Row	Col	Row	Col	Row	Col	Row	Col	Row
BA1	BA1	A22	A22	A23	A23	A24	A24	A25	A25	A26	A26
BA0	BA0	A21	A21	A22	A22	A23	A23	A24	A24	A25	A25
MA13	MA13	—	—	—	—	—	—	—	—	—	A24
MA12	MA12	—	—	—	—	—	A22	—	A23	—	A23
MA11	MA11	—	A20	—	A21	—	A21	—	A22	—	A22
MA10	MA10	—	A19	—	A20	—	A20	—	A21	—	A21
MA9	MA9	—	A18	—	A19	—	A19	A10	A20	A10	A20
MA8	MA8	—	A17	A9	A18	A9	A18	A9	A19	A9	A19
MA7	MA7	A8	A16	A8	A17	A8	A17	A8	A18	A8	A18
MA6	MA6	A7	A15	A7	A16	A7	A16	A7	A17	A7	A17
MA5	MA5	A6	A14	A6	A15	A6	A15	A6	A16	A6	A16
MA4	MA4	A5	A13	A5	A14	A5	A14	A5	A15	A5	A15

Table 30-24. Address Multiplexing by Column/Row Width for 16-bit Devices (continued)

Device Pins	ESDCTL Pins	16-bit LPDDR Memory Device									
		64 Mbytes		128 Mbytes		256 Mbytes		512 Mbytes		1 Mbytes	
		8 col 12 row		9 col 12 row		9 col 13 row		10 col 13 row		10 col 14 row	
		Col	Row	Col	Row	Col	Row	Col	Row	Col	Row
MA3	MA3	A4	A12	A4	A13	A4	A13	A4	A14	A4	A14
MA2	MA2	A3	A11	A3	A12	A3	A12	A3	A13	A3	A13
MA1	MA1	A2	A10	A2	A11	A2	A11	A2	A12	A2	A12
MA0	MA0	A1	A9	A1	A10	A1	A10	A1	A11	A1	A11

Table 30-25. Address Multiplexing by Column/Row Width for 32-bit Devices

Device Pins	ESDCTL Pins	32-bit LPDDR Memory Device											
		64 Mbytes		128 Mbytes		256 Mbytes		512 Mbytes		1 Gbytes		2 Gbytes	
		8 col 11 row		8 col 12 row		8 col 13 row		9 col 13 row		9 col 14 row		10 col 14 row	
		Col	Row	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row
BA1	BA1	A22	A22	A23	A23	A24	A24	A25	A25	A26	A26	A27	A27
BA0	BA0	A21	A21	A22	A22	A23	A23	A24	A24	A25	A25	A26	A26
MA13	MA13	—	—	—	—	—	—	—	—	—	A24	—	A25
MA12	MA12	—	—	—	—	—	A22	—	A23	—	A23	—	A24
MA11	MA11	—	—	—	A21	—	A21	—	A22	—	A22	—	A23
MA10	MA10	—	A20	—	A20	—	A20	—	A21	—	A21	—	A22
MA9	MA9	—	A19	—	A19	—	A19	—	A20	—	A20	A11	A21
MA8	MA8	—	A18	—	A18	—	A18	A10	A19	A10	A19	A10	A20
MA7	MA7	A9	A17	A9	A17	A9	A17	A9	A18	A9	A18	A9	A19
MA6	MA6	A8	A16	A8	A16	A8	A16	A8	A17	A8	A17	A8	A18
MA5	MA5	A7	A15	A7	A15	A7	A15	A7	A16	A7	A16	A7	A17
MA4	MA4	A6	A14	A6	A14	A6	A14	A6	A15	A6	A15	A6	A16
MA3	MA3	A5	A13	A5	A13	A5	A13	A5	A14	A5	A14	A5	A15
MA2	MA2	A4	A12	A4	A12	A4	A12	A4	A13	A4	A13	A4	A14
MA1	MA1	A3	A11	A3	A11	A3	A11	A3	A12	A3	A12	A3	A13
MA0	MA0	A2	A10	A2	A10	A2	A10	A2	A11	A2	A11	A2	A12

30.8.1.1.1 Bank Interleaving

The controller supports bank interleaving mode. This mode can be enabled through a configuration bit in the ESDMISC register. In this mode, the address decoding is different because the row and bank location in the AXI address switch. The affect of this change is that when the end of a certain row in write or read operation is reached, the next row is from another bank. Therefore, the controller can prepare the new row earlier, which increases performance.

30.8.1.2 Bank Addresses

The controller supports either 4 or 8 bank devices. All connected devices should be either 4 or 8 banks devices, but not mixed. A configuration bit exist for these two modes in ESDMISC register.

The bank decoding from the AXI address depends on various values, as follows:

- Device width ×16/×32
- Column size
- Row size, for bank interleaving off mode.

See Table 30-23 for adecoding example (for bank interleaving off mode).

30.8.2 Refresh

Enhanced SDRAM Controller hardware satisfies all refresh requirements after an initial configuration by the user software. Zero, 1, 2, 4, 8, or 16 refresh cycles are scheduled at 31.25 µs (nominal 32 kHz clock) intervals, providing 0, 2048, 4096, 8192, 16384, or 32768 refresh cycles every 64 ms. The refresh rate is programmed through the REFR field in the ESDCTL0 and ESDCTL1 registers. Each array can have a different rate, allowing a mix of SDRAM/LPDDR devices, or different SDRAMs density. Refresh is disabled by hardware reset.

A refresh request is made pending at each rising edge on the 32 kHz clock. In response to this request, the hardware gains control of the memory device as soon as any in-process bus cycle completes. Once it has gained control of the memory, commands are issued to precharge all banks. Following a row precharge delay (t_{RP}), an auto-refresh command is issued. At t_{RFC} intervals, additional auto-refresh cycles are issued until the specified number of cycles have been run.

Figure 30-29 illustrates a 2 refresh sequence. Burst transfers in progress when the refresh request arrives are allowed to complete prior to the refresh operation. Memory device bus accesses queued after the refresh request are held off until the refresh completes.

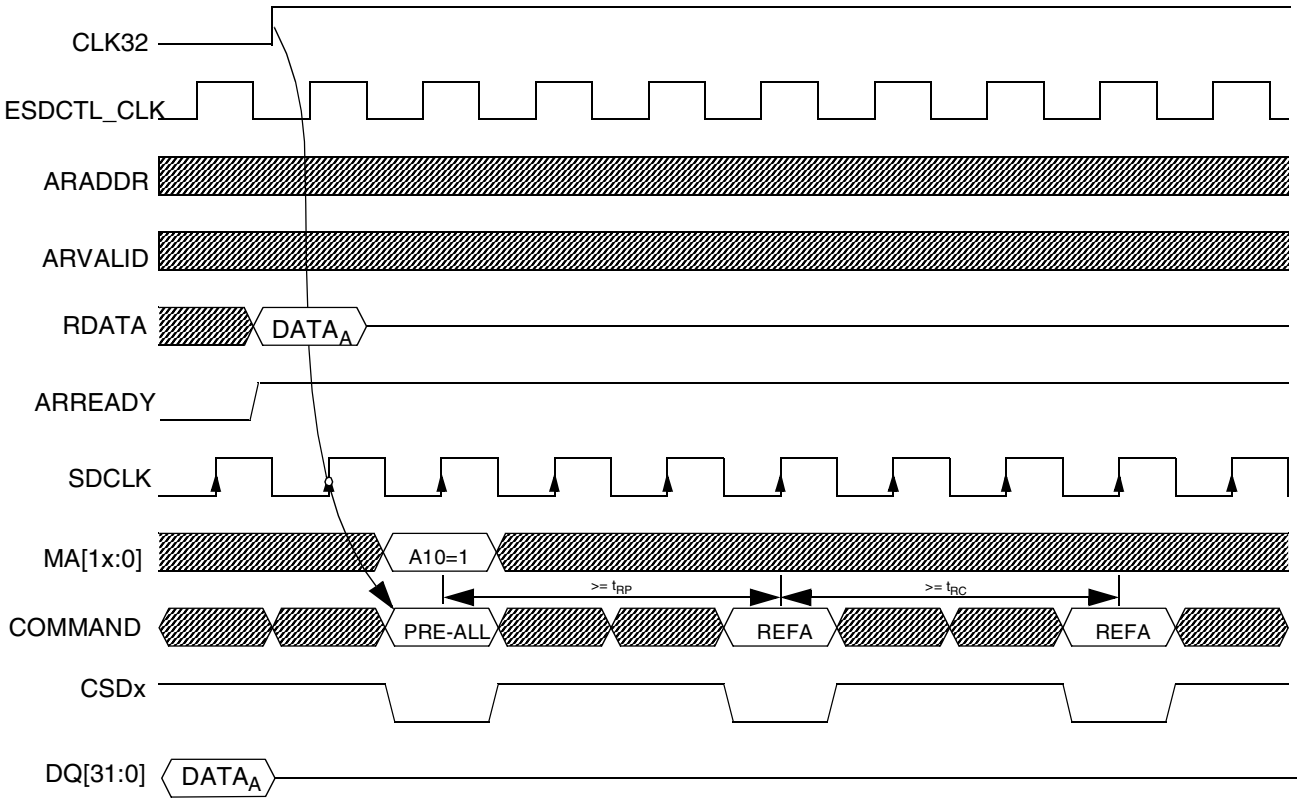


Figure 30-29. Hardware Refresh Timing Diagram

In Figure 30-30, an access is queued just as the refresh begins. This cycle is delayed until the precharge and single refresh (REFR=01) cycles are run. Bus cycles targeted to other memory or peripheral devices are allowed to progress normally while the refresh is in progress. None of the pins shared between the SDRAM and other devices are required for the refresh operation.

NOTE

Since REFRESH commands (requires all banks to be in IDLE state, achieved by PRECHARGE ALL) are issued automatically by the Enhanced SDRAM Controller at each 32 kHz clock, address bits A10 (for both 16 and 32-bit devices) cannot be shared with another peripheral's address bus in the system.

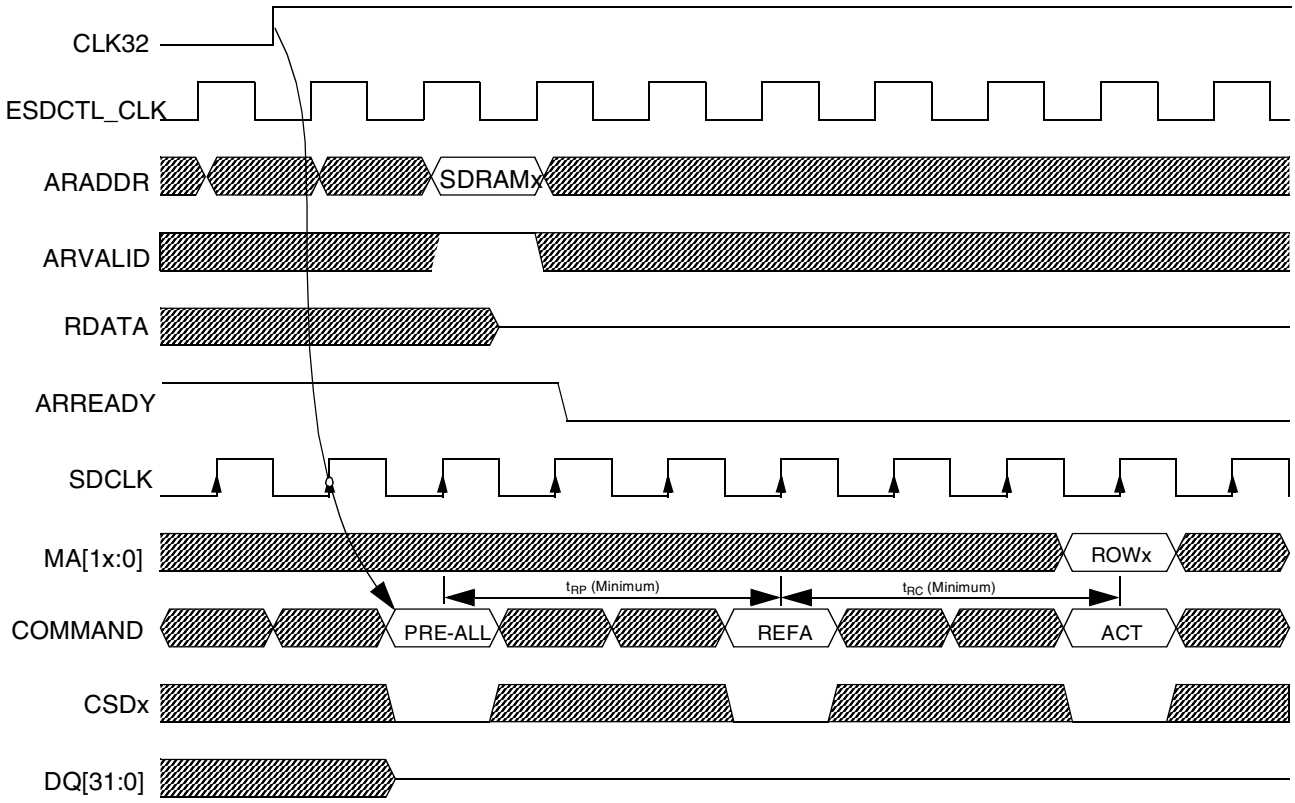


Figure 30-30. Hardware Refresh with Pending Bus Cycle Timing Diagram

30.8.3 Low-Power Operating Modes

This section describes the low-power operating modes of the Enhanced SDRAM Controller as a functions of the various memory devices. Table 30-26 lists and summarizes the low power modes supported by the Enhanced SDRAM Controller.

Table 30-26. ESDRAMC Low Power Operating Modes

System Operating Mode	Memory Device Low Power Operating Mode	WakeUp Penalty
RUN	POWER DOWN MODE	tXP
RUN	PRECHARGE Banks	1 clock cycle
RUN	MANUAL SELF REFRESH MODE	tXS + 2 Refresh Period
STOP	SELF REFRESH MODE	tXS + 2 Refresh Period

30.8.3.1 Self-Refresh Mode for DDR2/LPDDR Devices

This operating mode (see Figure 30-31 and Figure 30-32) allows the software/user to control a self-refresh mode entry of the external memory device if refresh has been enabled, during system run mode. When this mode is selected (using the special command register) and refresh is enabled, the Enhanced SDRAM Controller completes any active access, and a self-refresh command to the external device is issued. No

access is allowed to the respective CSD during manual self-refresh mode. If refresh has not been enabled, the Enhanced SDRAM Controller places the memory in a low-power consumption mode known as power down. The LPACK signal (low-power mode acknowledge) is not asserted if only one CSD enters manual self-refresh mode.

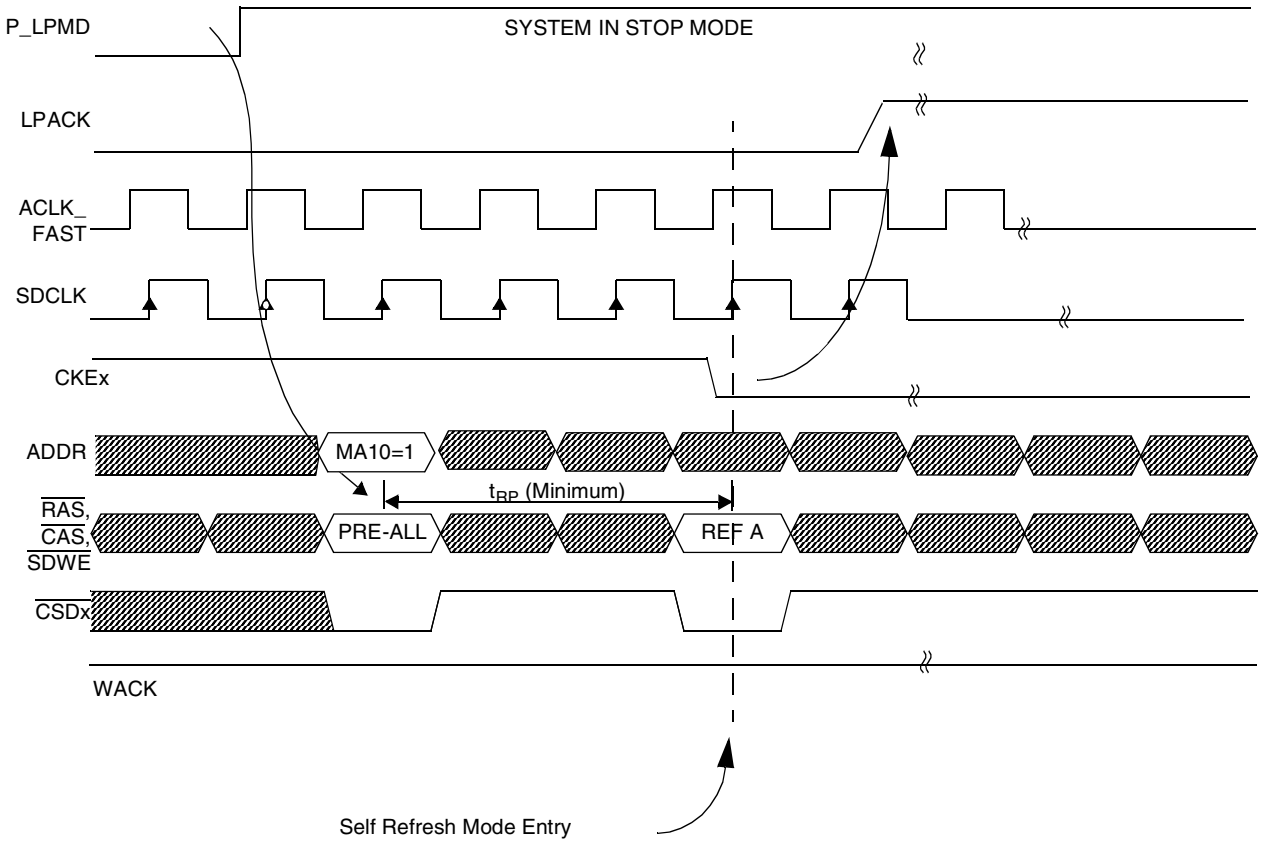


Figure 30-31. Enter Self-Refresh Mode during System Stop Mode

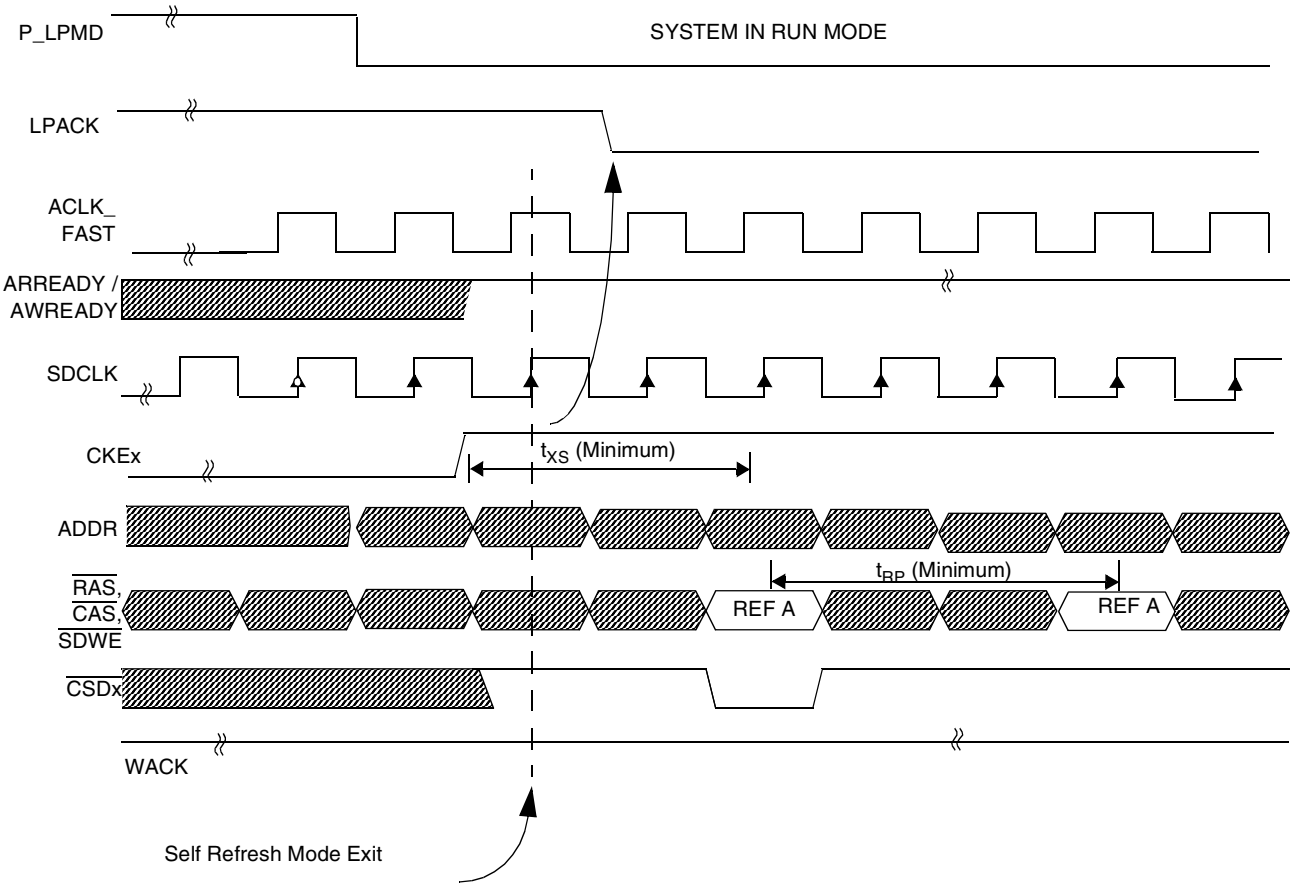


Figure 30-32. Exit Self-Refresh Mode during System StOP Mode

30.8.3.2 Manual Self-Refresh Mode for DDR2/LPDDR Devices

This operating mode allows the software/user to control a self-refresh mode entry of the external DDR2/LPDDR device if refresh has been enabled, during system run mode. When this mode is selected (using the special command register) and refresh is enabled the Enhanced SDRAM Controller will complete any active access and a self-refresh command to the external device will be issued. No access is allowed to the respective CSD during manual self-refresh mode. If refresh has not been enabled, the Enhanced SDRAM Controller places the memory in a low-power consumption mode known as power down. The LPACK signal (low-power mode acknowledge) will not be asserted if only one CSD enters manual self refresh mode.

NOTE

A manual precharge all should be initiated by the user before a manual self refresh.

To exit manual self-refresh mode, a “resume normal operation” command must be issued in the special command register. When the command is issued, the controller exits the SDRAM device from self-refresh mode and issues auto-refresh cycles (if the refresh has been enabled).

Figure 30-33 and Figure 30-34 show the timing information.

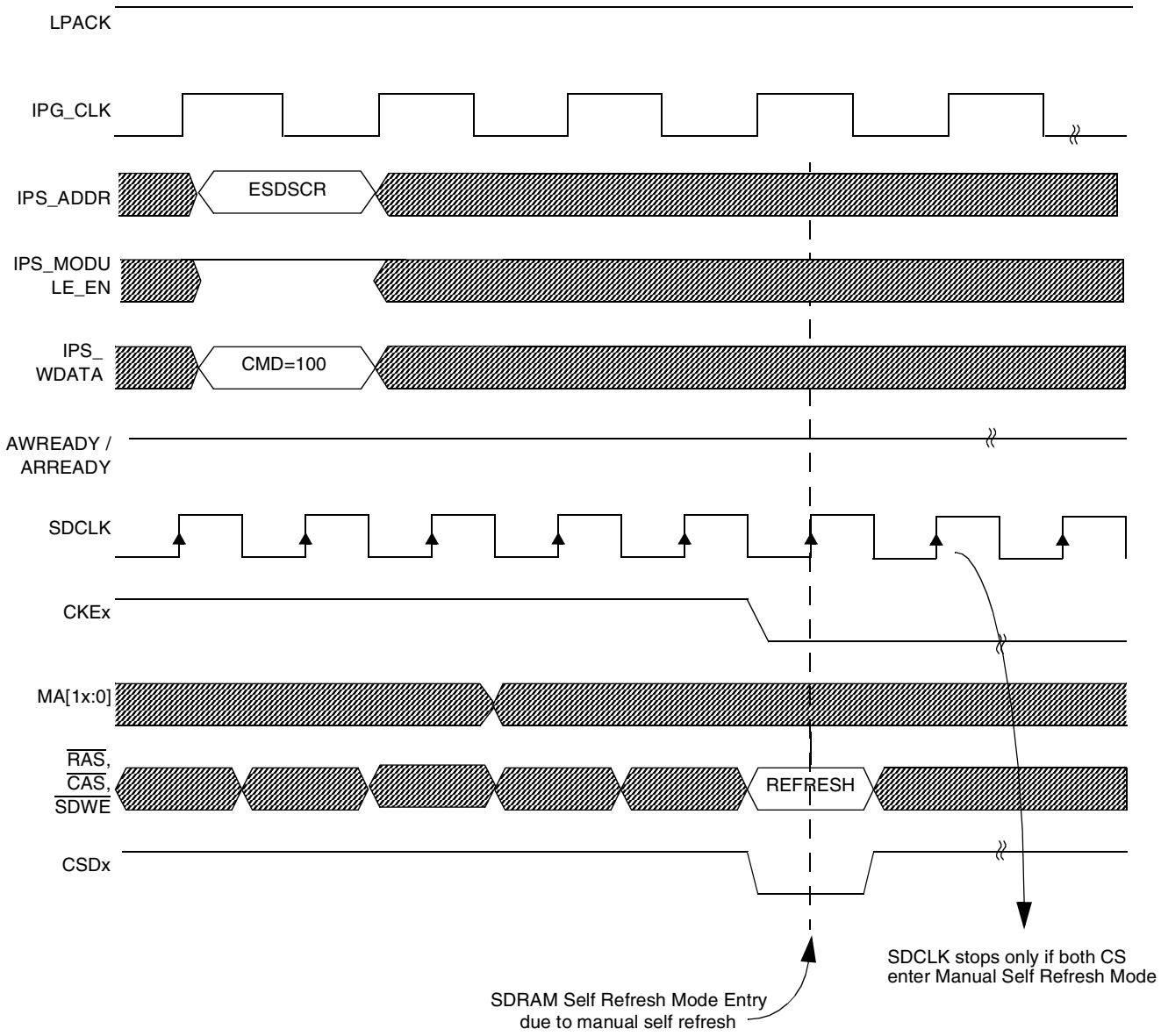


Figure 30-33. Manual Self Refresh Entry Timing Diagram

NOTE

SDCLK stops, only if both chip selects are in manual self-refresh. This allows the usage of one chip select while the other is in manual self-refresh (in case both chip selects are in use).

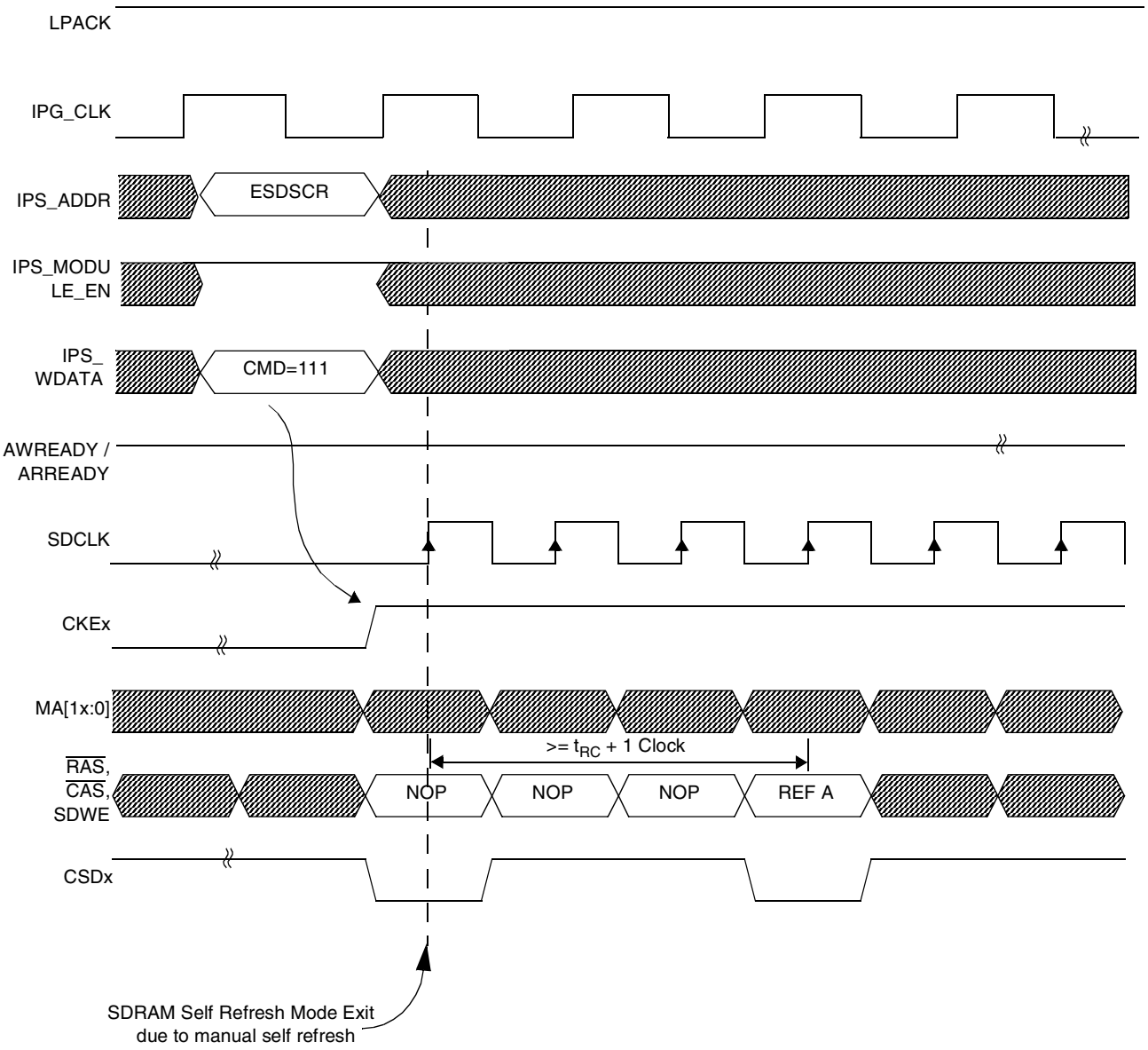


Figure 30-34. Manual Self -Refresh Exit Timing Diagram

30.8.3.3 Precharge Power-Down Mode

All the low power operating mode described in the above paragraphs will be activated only if the system enters low power operating mode, for example, stop mode. The Enhanced SDRAM Controller has the capability to reduce power consumption if the DDR2/LPDDR memory utilization is low, by setting the SDRAM device in power-down mode. This mode is activated through the PWDT bits in the ESDCTL0 and/or ESDCTL1 registers. During this operating mode the ESDRAMC automatically issues the REFRESH commands toward the LPDDR memories at the rate defined by the SREFR bits in the ESDCTL0 and/or ESDCTL1 registers.

Programming PWDT[1:0] = 01 causes the Enhanced SDRAM Controller to place the memories in power down mode anytime the controller detects that no banks are active. This mode is useful in applications where a memory array is accessed infrequently and the chances of another access to the same page are minimal.

Reading or writing to memory activates a page within the addressed bank. Reset, software generated precharge, and hardware initiated refresh are three ways to close an active bank. The periodically occurring refresh will be the normal means that invokes the power down mode. At each refresh interval, all banks will be closed by a precharge-all command, followed by the refresh operation. The controller will then issue the power down command to the memories. A few cycle delay is incurred with the first read or write cycle in order to restart the clocks, but only on the first cycle. After that, the clocks will continue to run until the next refresh operation or until any active banks are manually precharged.

Page misses on read and write cycles cause the addressed bank to be closed (precharged) and a new page opened within the bank. This operation does not cause the clocks to stop, nor does manually precharging only a single bank within the memory. All banks within the memory must be inactive before the power down mode is invoked.

Power-down mode occurs if CKE is registered low coincident with a NOP or command inhibit when no accesses are in progress. Entering power down deactivates the input and output buffers (excluding CKE) of the device. Power-down modee is exited by registering a NOP or command inhibit and CKE high at the desired clock edge.

For LPDDR SDRAM, Figure 30-35 and Figure 30-36 illustrate the power-down mode entry and exit respectively.

NOTE

Since the ESDRAMC does npt issue AUTO PRECHARGE commands toward the SDRAM, the software must issue a PRECHARGE ALL command in order to enter Precharge Low Power Down Mode, to wait for the PRECHARGE timer (PRCT) to close/precharge all active banks, or to wait for the next REFRESH cycle in order to enter this low-power mode. (During the REFRESH cycle, the ESDRAMC automatically issues the PRECHARGE ALL command).

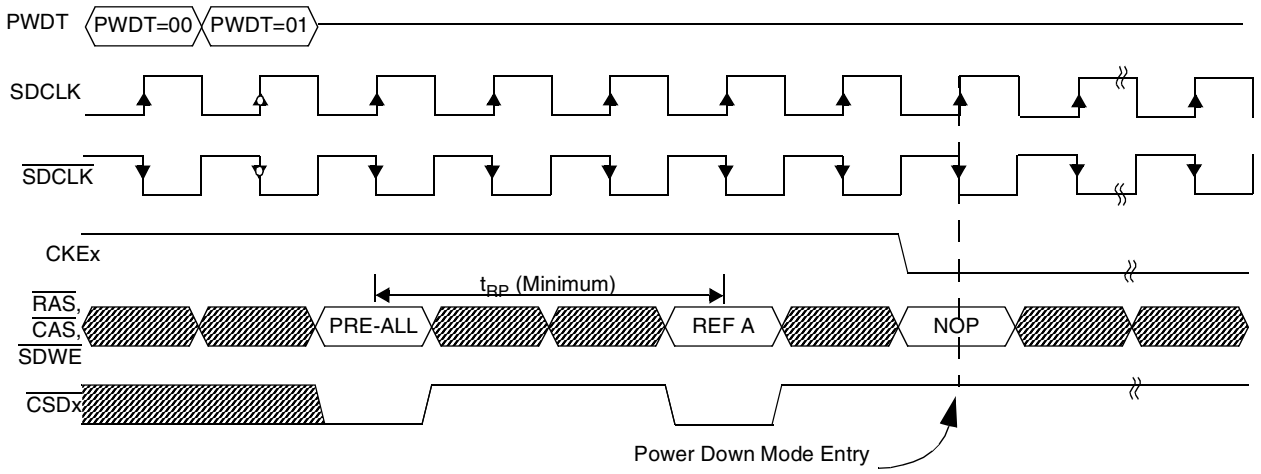


Figure 30-35. Mobile DDR SDRAM Precharge Power Down Mode Entry Timing Diagram

Power-down mode for several mobile/low-power DDRs requires that the clock CK (and $\overline{CK}$) continue running. The PWR CK EN (power down clock enable for mobile/low power DDR SDRAM) should be set to 1 to enable this option.

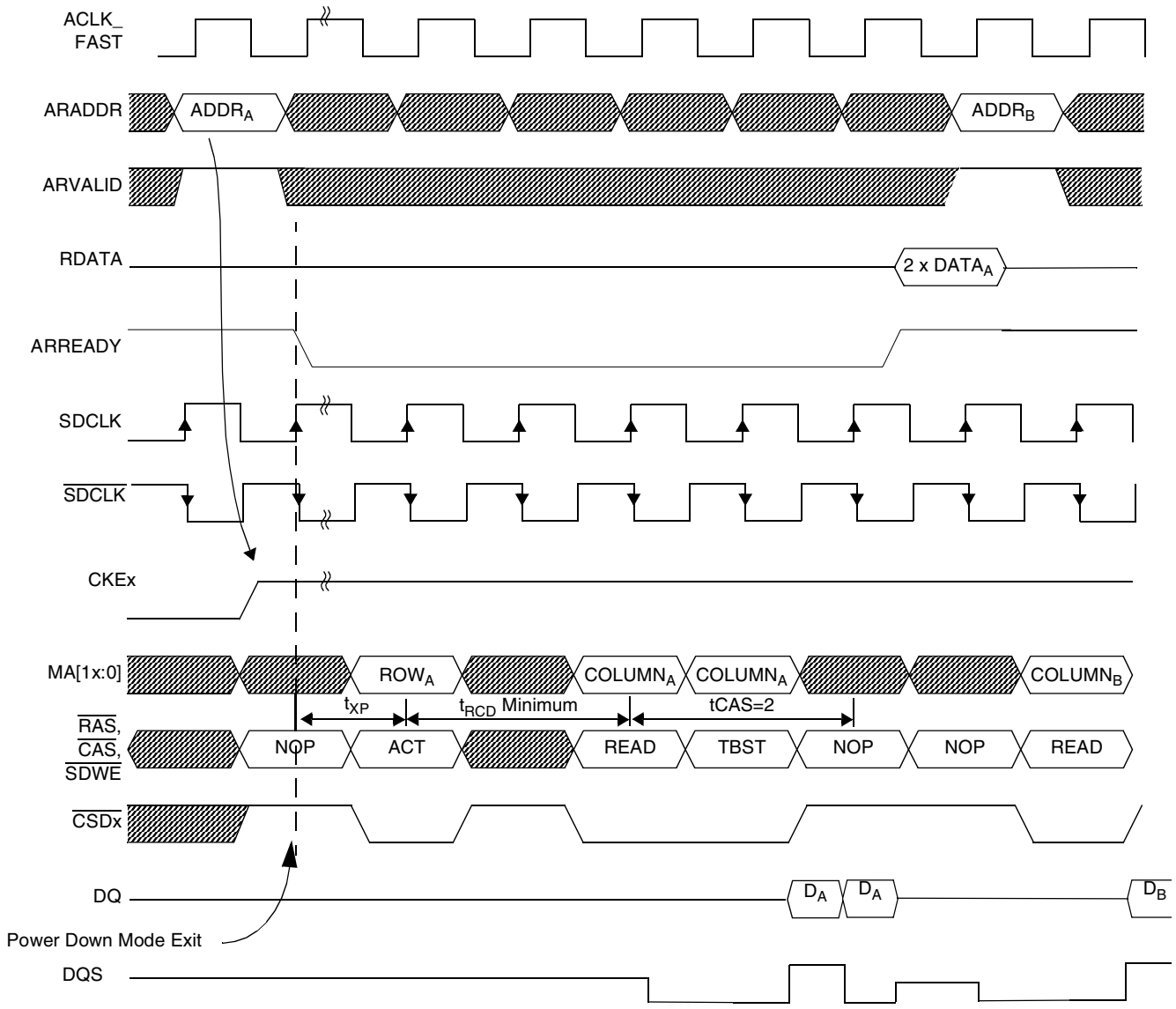


Figure 30-36. Mobile DDR SDRAM Precharge Power Down Mode Exit Timing Diagram

30.8.3.4 Active Power-Down Mode

The second clock suspend mode is selected whenever PWDT[1:0] = 1x. In this mode the SDCLK is stopped after a selectable delay from the last access to the array. Active banks are not closed prior to disabling the LPDDR clock. Either 64 (PWDT[1:0] = 10) or 128 (PWDT[1:0] = 11) cycle delays are possible. LPDDR clocks are counted from the end of the last read or write access. Subsequent read or write accesses and self-refresh modes reset the counter. Auto-refresh cycles do not affect the counter; however, if the counter expires during a refresh operation the clock will be disabled immediately following the refresh.

The distinguishing factor between precharge power-down mode and active power-down mode is whether banks remain active while the clock is stopped. Active power-down allows banks to remain activated while precharge power-down does not.

Figure 30-37 illustrates LPDDR SDRAM Active Power Down Mode entry and exit timing diagram.

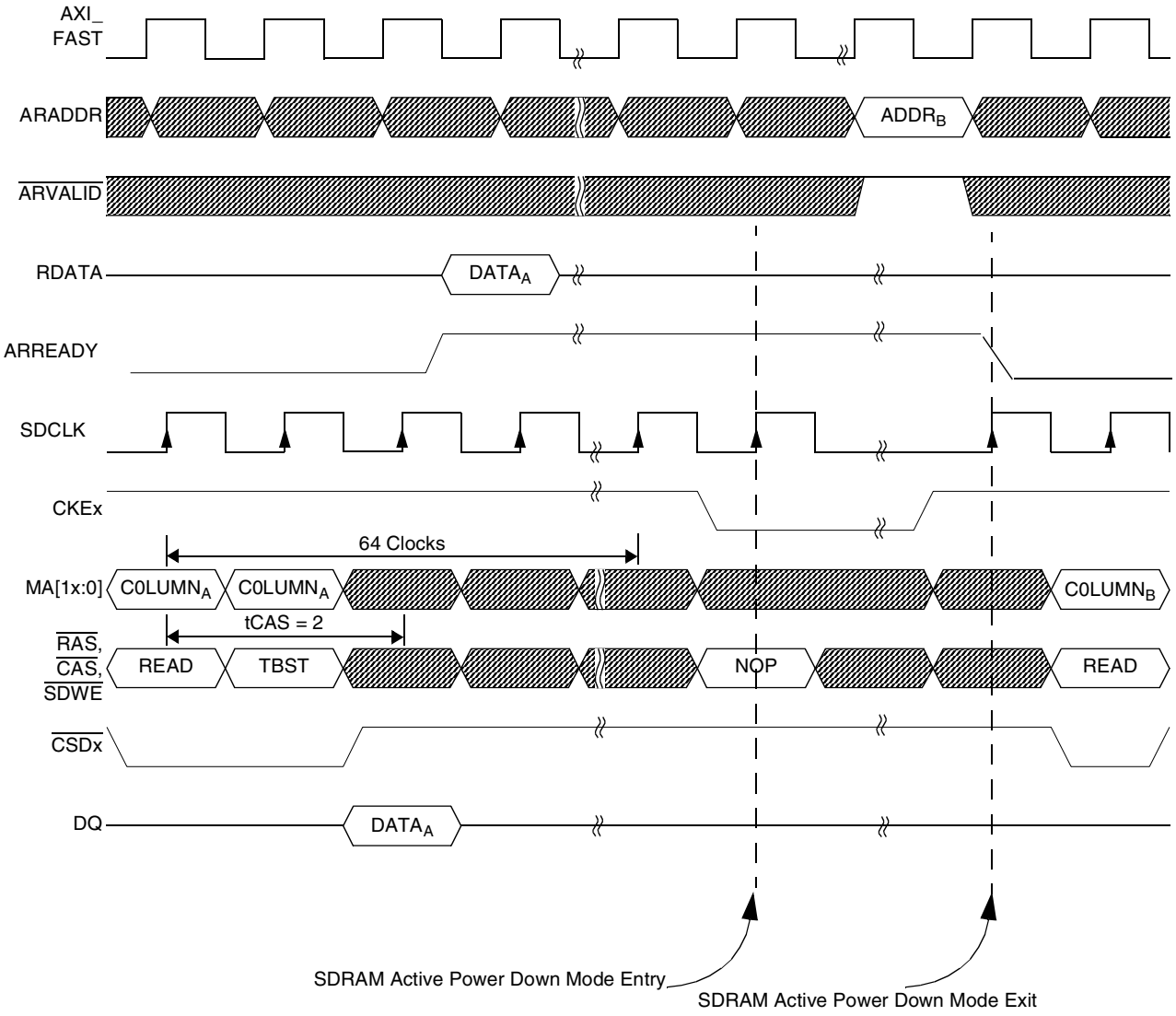


Figure 30-37. Mobile DDR SDRAM Active Power Down Mode Timing Diagram

Power-down mode for several mobile DDRs require that the clock CK (and $\overline{CK}$) continue running. The PWR CK EN (power-down clock enable for mobile DDR SDRAM) should be set to “1” to enable this option.

30.8.3.5 Precharge Bank(s)—Low Power Mode

Closing (due to precharge command) the last used/open row in any non-active bank within a chip select reduces the power consumption of the external memory device. The power saving is device dependent,

and one should consult/examine the external memory device specification for more details on power consumption reduction.

The precharge bank is activated if PRCT is enabled. A precharge command is issued after 2 × PRCT clocks (ESDCTL_CLK, up to 200 MHz) of no activity (as shown in Table 30-22) to one of the banks. The number of cycles before the precharge command is issued depends on command bus (WE, RAS, CAS and CSD) availability (meaning there is no active access to other bank) and the memory timing parameters.

30.8.3.6 LPDDR Frequency Change

The following steps need to be performed prior to a frequency change in a LPDDR based system, in order for the DDRC delay line recalibration locking.

The frequency change request signal (from system) indicates that the ESDCTL clock is about to change. This signal is considered asynchronous and is synchronized internally. When this signal is asserted, the current ongoing accesses finish and all pending accesses are put on hold. Then the ESDCTL performs self-refresh enter to the DDR SDRAM memory and finally asserts the FREQ_CHNG_ACK signal, indicating the frequency is allowed to change. The FREQ_CHNG_REQ should be negated only after the frequency change was applied. The ESDCTL performs self-refresh exit to the DDR memory. In DDR mode, the ESDCTL also updates the delay lines to the new frequency that is used to capture the DDR data.

30.8.4 Command Encoding

Table 30-27 summarizes the command encoding utilized by this controller. These commands represent a subset of the commands defined by the JEDEC standard.

Table 30-27. LPDDR Command Encoding

Function	Symbol	CKE n-1	CKE n	CS	RAS	CAS	WE	A11	A10	BA[1:0]	A[13:0]
Deselect	DSEL	H	X	H	X	X	X	X	X	X	X
No Operation	NOP	H	X	L	H	H	H	X	X	X	X
Read	READ	H	X	L	H	L	H	V	L	V	V
Write	WRIT	H	X	L	H	L	L	V	L	V	V
Bank Activate	ACT	H	X	L	L	H	H	V	V	V	V
Burst Terminate[1]	TBST	H	X	L	H	H	L	X	X	V	X
Precharge Select Bank	PRE	H	X	L	L	H	L	V	L	V	X
Precharge All Banks	PALL	H	X	L	L	H	L	X	H	X	X
Auto-Refresh	CBR	H	X	L	L	L	H	X	X	X	X
Self Refresh Entry	SLFRSH	H	L	L	L	L	H	X	X	X	X
Self Refresh Exit	SLFRSHX	L	H	H	X	X	X	X	X	X	X
Deep Power-Down Entry	PWRDN	H	L	L	H	H	L	X	X	X	X
Deep Power-Down Exit	PWRDNX	L	H	L	H	H	H	X	X	X	X

Table 30-27. LPDDR Command Encoding (continued)

Function	Symbol	CKE n-1	CKE n	CS	RAS	CAS	WE	A11	A10	BA[1:0]	A[13:0]
Power-Down Entry	PWRDN	H	L	X	X	X	X	X	X	X	X
Power-Down Exit	PWRDNX	L	H	H	X	X	X	X	X	X	X
Mode Register Set[2]	MRS	H	X	L	L	L	L	L	L	V	V

[1]For Mobile DDR, applies only to read bursts (with auto precharge disabled).
[2]BA0-BA1 select either the mode register, the extended mode register or the LOW POWER extended mode register.

The unsupported JEDEC commands are read and write with auto precharge (when once the read/write is complete the active row in the used bank is precharged automatically).

30.8.4.1 Reset

Assertion of the $\overline{\text{RST}}$ signal initializes the controller into the idle state and disables the module. While disabled, the controller remains in the idle state with the internal clocks stopped. The reset state of the control register allows for basic read/write operations sufficient to fetch the reset vector and execute the initialization code. A complete initialization of the controller should be performed as part of the start-up code sequence.

Read/write cycles, refresh, and low-power mode requests as well as power-down time-outs all trigger transitions out of the idle state. As shown in the simplified enhanced SDRAM controller state diagram pictured in Figure 30-38, state transitions due to a read or write request depend on the operating mode. Other transitions require the corresponding function to be enabled in the ESDCTL registers. Some state transitions have been removed from the figure to minimize complexity and allow an easier understanding of the basic controller operation.

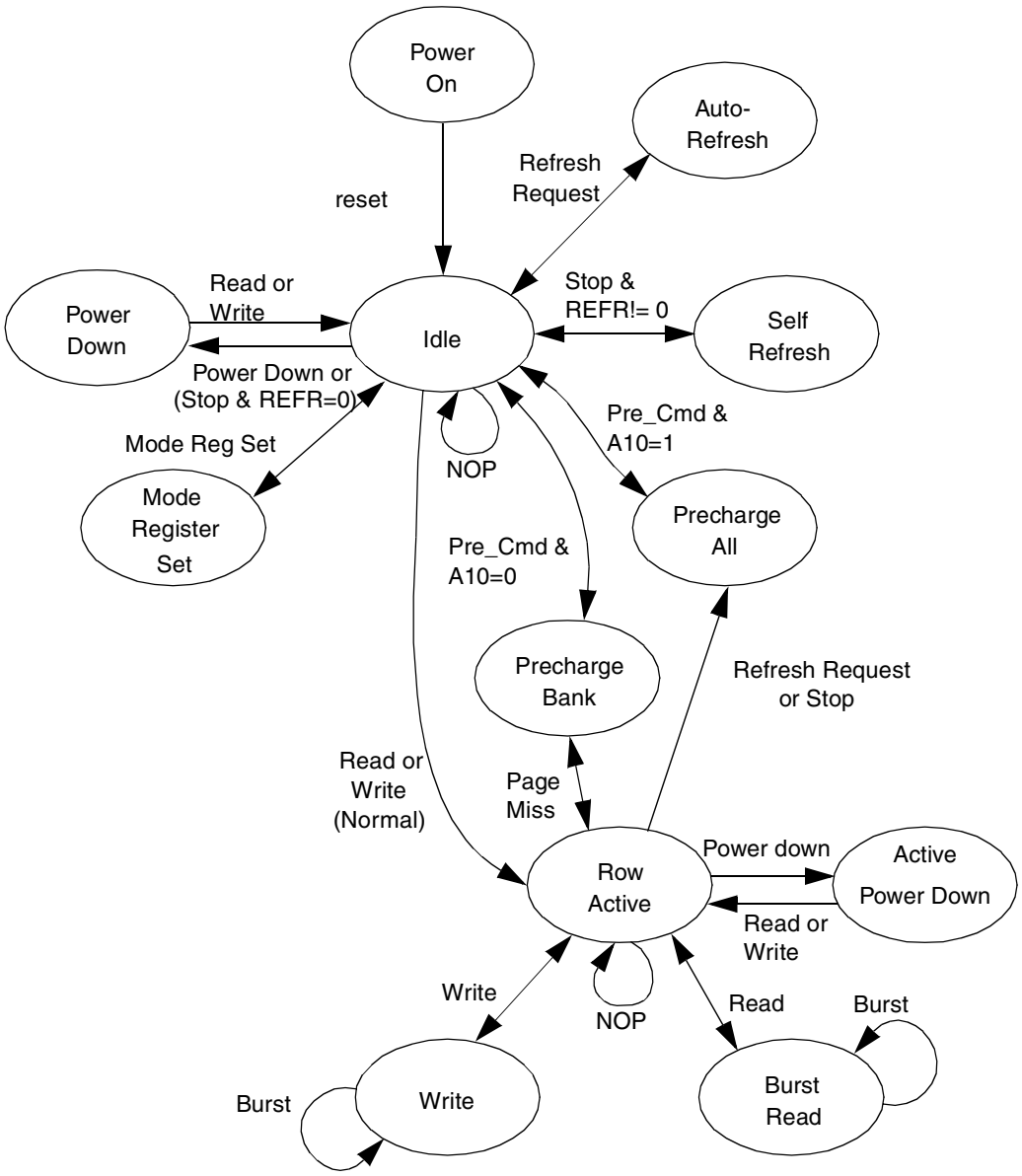


Figure 30-38. Simplified Enhanced SDRAM Controller State Diagram

The following subsections document the operation of each of the operating modes.

30.8.4.2 Warm Reset

This section describes the flow of getting into self-refresh mode before and during warm reset, as well as configuration that is needed to be made to ESDCTL when getting out of warm reset.

Please note that after issuing a precharge all via special command register in the exit warm reset, it is recommended by devices to issue a manual auto refresh command, but not necessary.

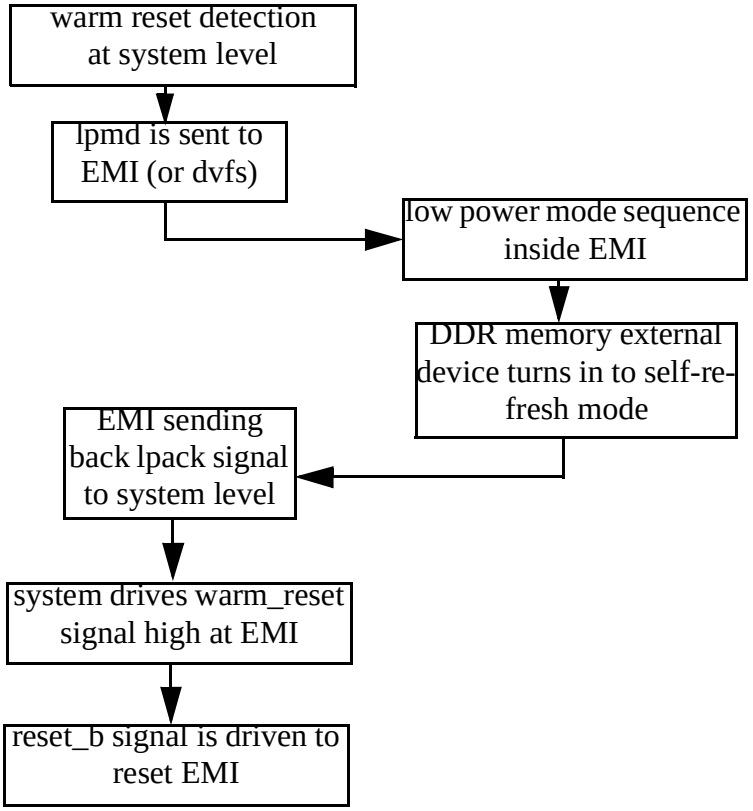


Figure 30-39. Entering into Self-Refresh before warm_reset Assertion

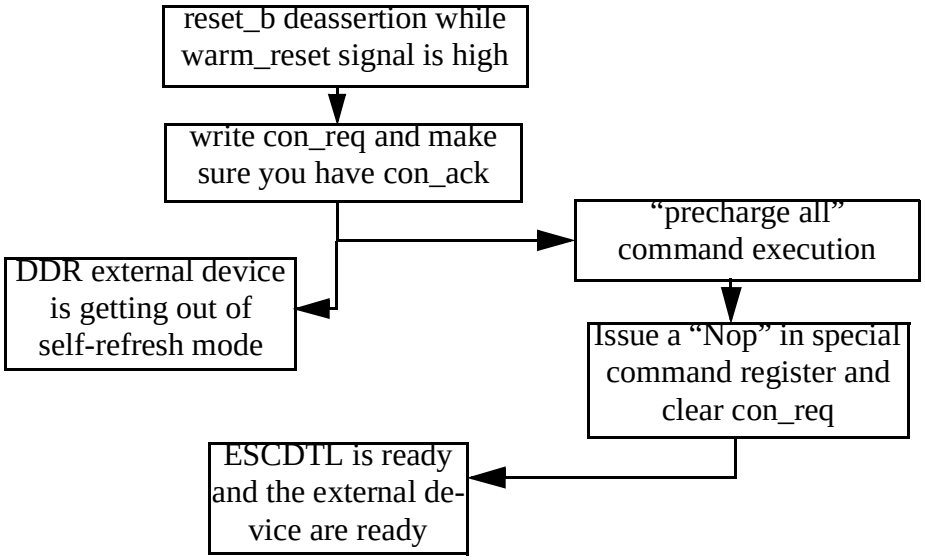


Figure 30-40. Sequence of Exiting Self-Refresh after Warm Reset

30.8.5 ODT behavior

The two type of terminations are as follows:

- Termination to the DQS/DQ I/O pads—this is controlled through the TERM_CTL configuration fields in ESDMISC register. This termination is working while there are no write accesses. It is turned off when write to external DRAM is done. Through its configuration fields it is possible to disable it or change termination resistance. This is done per byte lane.
- Termination to external DRAM. This can be enabled through ODT_EN bit in ESDMISC register.

When enabled the controller enables the termination in the DRAM that is not being accesses.

30.9 Initialization Information

The following paragraphs provide details on selecting compatible SDRAM memories and configuring the controller to work with the memory system.

30.9.1 Memory Device Selection

Many DDR2/mDDR memory devices types are supported by the Enhanced SDRAM Controller. Important characteristics to consider when choosing a memory device are as follows:

- Device size limit is 256 Mbytes per chip select
- Support 4 banks mDDR memory devices
- Support 4 or 8 banks DDR2 memory devices
- Support memory devices of up to 200 MHz
- Support memory devices with CAS latency of 3
- Support data width of ×16 or ×32

NOTE

Some DDR2 devices support higher frequency than 200 MHz and require higher CAS latency than 3. But this devices most often support CAS 3 latency when lowering the frequency to 200 MHz. The CAS latency is configurable in memory devices.

30.9.2 CAS Latency

The controller supports only CAS latency of 3 cycles. This is the required latency for DDR2-400 and LPDDR-400 devices.

30.9.3 LPDDR/DDR2 Initialization Sequence

On power up ESDCTL is disabled. The SDE bit must then be enabled first, and it keeps 400 µs of idle period. During the first 200 µs ‘CKE’ signal to memory is low, and during the second 200 µs ‘CKE’ signal to memory is high. This is to comply with both the DDR2 and mDDR initialization sequence.

The initialization of the memory device should be done according to JEDEC standard or specific memory device requirement.

Memory device initialization is done via ESDSCR register (see Table 30-13) by choosing desired command and selecting what chip select, bank, and address is to be sent to memory device.

For example, to issue precharge-all command to CS0, set the following values on the ESDSCR register:

- CON_REQ = 1 (this is required through all the init sequence).
- BA= 0x0
- CS = 0x0
- CMD = 0x1
- PSEUDO_ADDR = 0x400

In DDR2 there is a 200 cycle delay that is required after resetting memory device DLL. This delay can be done automatically or manually, as follows.

- For automatic delay, select ‘AUTO_DLL_PAUSE’ (see Table 30-12). The controller identifies the DLL reset that was sent to the memory and keeps the needed delay. This is the default option.
- For manual delay, select ‘MAN_DLL_PAUSE’ (see Table 30-13) along with the needed ‘LMR’ command selection.

Other guidelines are as follows:

- CON_REQ bit on ESDSCR register should remain high until all initialization is over. Lowering it is allowed only in the last initialization command.
- The following functions should not be enabled before the memory initialization is over.
  - SREFR field on ESDCTL0 and ESDCTL1 register
  - PWDT and SRT fields on ESDCTL0 and ESDCTL1 registers
  - PRCTn field on ESDPRCT0 and ESDPRCT1 registers
  - ODT_EN field on ESDMISC registers

The init sequence structure uses the following steps:

1. Enable the SDE bit on the ESDCTL register.
2. If using DDR2, enable the DDR2_EN bit in the ESDMISC register.
3. Make a sequence of commands to initialize the external memory. Every command is issued though the ESDSCR register. The user must set the required command and the matching values on the CS, BA, and MA signals.
4. The controller issues the selected command and signal values to the external memory. Note that the CON_REQ bit must be kept high during this process.
5. At this point, perform all remaining desired configurations to the controller.

30.9.3.1 mDDR Initialization Example

The following example code demonstrates how to initialize chip select 0 and 1.:

Example 30-1. Initializing Chip Select 0 and 1

```
setmem ESDCTL0 = 0x83220000  // ESDCTL0: Enable controller
// Init DRAM on CS0
setmem ESDSCR = 0x04008008  // ESDSCR: Precharge command
setmem ESDSCR= 0x00008010  // ESDSCR: Refresh command
setmem ESDSCR= 0x00008010  // ESDSCR: Refresh command
setmem ESDSCR= 0x00338018  // ESDSCR: LMR with CAS=3 and BL=3 (Burst Length = 8)
setmem ESDSCR= 0x0020801a  // ESDSCR: EMR with half Drive strength
setmem ESDCTL0 = 0xC3220000  // ESDCTL0: 14 ROW, 10 COL, 32Bit, SREF=8
// ESDCFG0: tRFC:22clks, tXSR:28clks, tXP:2clks, tWTR:2clk, tRP:3clks, tMRD:2clks
//          tRAS:8clks, tRRD:2clks, tWR:3clks, tRCD:3clks, tRC:11clks
setmem ESDCFG0 = 0xC33574AA
setmem ESDMISC = 0x000a1700  // ESDMISC: AP=10, Bank interleaving on, MIF3 en, RALAT=2
setmem ESDCTL1 = 0x83220000  // ESDCTL1: Enable controller
//// Init DRAM on CS1
setmem ESDSCR = 0x0400800c  // ESDSCR: Precharge command
setmem ESDSCR= 0x00008014  // ESDSCR: Refresh command
setmem ESDSCR = 0x00008014  // ESDSCR: Refresh command
setmem ESDSCR = 0x0033801c  // ESDSCR: LMR with CAS=3 and BL=3 (Burst Length = 8)
setmem ESDSCR = 0x0020801e  // ESDSCR: EMR with half Drive strength
setmem ESDCTL1= 0xC3220000  // ESDCTL1: 14 ROW, 10 COL, 32Bit, SREF=8
//// ESDCFG1: tRFC:21clks, tXSR:28clks, tXP:2clks, tWTR:2clk, tRP:3clks, tMRD:2clks
////          tRAS:8clks, tRRD:2clks, tWR:3clks, tRCD:3clks, tRC:11clks
setmem ESDCFG1 = 0xC33574AA
setmem ESDSCR = 0x00000000 // ESDSCR - clear "configuration request" bit
```

Chapter 31
Enhanced Secured Digital Host Controller (eSDHC)

31.1 Overview

The eSDHC provides the interface between the host system and MMC/SD/SDIO/CE-ATA cards, including cards with reduced size or mini cards.

Figure 31-1 shows the eSDHC and its connections within the device. The eSDHC acts as a bridge, passing host bus transactions to the MMC/SD/SDIO/CE-ATA cards by sending commands and performing data accesses to and from the cards. It handles the MMC/SD/SDIO/CE-ATA protocols at the transmission level.

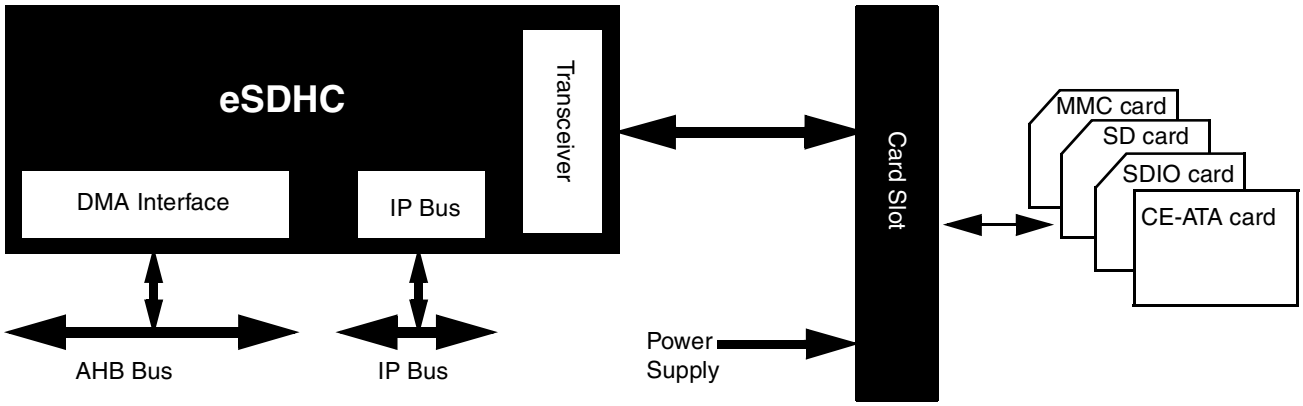


Figure 31-1. eSDHC System Connection

The different cards supported by the eSDHC are described as follows:

- MultiMediaCard (MMC)

  This is a universal low-cost data storage and communication media that is designed to cover a wide area of applications including mobile video and gaming both on pre-loaded MMC cards and downloadable from cellular phone, WLAN or other wireless networks. Old MMC cards are based on 7-pin serial bus with a single data pin, while the newer high-speed MMC communication is based on an advanced 11-pin serial bus designed to operate at lower voltage.
- Secure Digital (SD) card

  This is an evolution of earlier MMC technology. It is specifically designed to meet the security, capacity, performance, and environment requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment and data transfer protocol are forward compatible with MMC, with some additions. Under the SD protocol, an SD card can be categorized as memory card, I/O card, or combo card (having both memory and I/O functions). The memory card invokes a copyright protection mechanism that complies with the SDMI security

standard. The I/O card provides high-speed data I/O with low power consumption for mobile electronic devices. For the sake of simplicity, Figure 31-1 does not show cards with reduced size or mini cards.

- Consumer electronics ATA (CE-ATA)

  This is a hard drive interface that is optimized for embedded applications. The device is layered on the top of the MMC protocol stack using the same physical interface. The interface electrical and signaling definition follows the MMC specification. For more details, see the CE-ATA Specification.

31.1.1 Features

The features of the eSDHC module include the following:

- Designed to work with MMC, MMC plus, MMC RS, SD memory, miniSD memory, SDIO, miniSDIO, SD Combo, and CE-ATA cards. Compatible with the following specifications:
  - MMC System Specification Version 4.2
  - SD Host Controller Standard Specification Version 2.0, including test event register support
  - SD Memory Card Specification Version 2.0: supports High-Capacity SD Memory Cards
  - SDIO Card Specification Version 2.0
  - CE-ATA Card Specification Version 1.0
- Supports 1, 4, or 8 bit MMC modes, 1bit or 4 bit SD and SDIO modes, and 1, 4, or 8 bit CE-ATA devices
  - Card bus clock frequency up to 52 MHz
  - Up to 416 Mbps of data transfer for MMC cards in 8-bit mode
  - Up to 200 Mbps of data transfer for SD/SDIO cards in 4-bit mode
  - Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes, also supports interrupt period
- Supports single-block and multi-block read and write
  - Block sizes of 1–4096 bytes
  - Supports pause during the data transfer at block gap
  - Supports Auto CMD12 for multi-block transfer
- Supports read/write features:
  - Write protection switch for write operations
  - SDIO read wait and suspend/resume operations
  - Includes a fully configurable 128 × 32-bit FIFO for read/write data
- Supports internal and external DMA capabilities
  - Supports Advanced DMA to perform linked memory access
- Supports both synchronous and asynchronous abort
- Host can initiate non-data transfer command while data transfer is in progress

31.1.2 Modes of Operation—Data Transfer Modes

The eSDHC can select the following modes for data transfer:

- MMC 1-, 4-, or 8-bit transfers in full-speed mode (up to 20 MHz) or high-speed mode (up to 52 MHz)
- SD 1- or 4-bit transfers in full-speed mode (up to 25 MHz) or high-speed mode (up to 50 MHz)
- CE-ATA 1-, 4-, or 8-bit transfers
- Identification Mode (up to 400 kHz)

31.2 External Signals

This section provides an overview of the external signals, including a block diagram and a table of the I/O signal properties.

31.2.1 Overview

Figure 31-2 shows the eSDHC I/O signals:

- SD_CLK is an internally-generated clock used to drive the MMC, SD, SDIO, or CE-ATA cards.
- CMD I/O is used to send commands and receive responses to/from the card.
- Eight data lines (DAT7–DAT0) are used to perform data transfers between the eSDHC and the card.
- SD_CD# and SD_WP are card-detection and write-protection signals directly routed from the socket. A low on SD_CD# indicates that a card is inserted, and a high on SD_WP indicates that the write-protect switch is active.
- SD_OD is an output signal generated at the SoC level outside eSDHC, and is used to select the external open-drain resistor.
- SD_LCTL is an output signal used to drive an external LED to indicate that the SD interface is busy.
- SD_VS is used to control the voltage at the SD_ pins listed above. When asserted, voltage is high (around 3.0V); when negated, voltage is low (around 1.8 V).

SD_CD#, SD_WP, SD_OD, SD_LCTL, and SD_VS are all optional for system implementation. If the eSDHC is desired to support a 4-bit data transfer, DAT7–DAT4 are optional and can be tied to high.

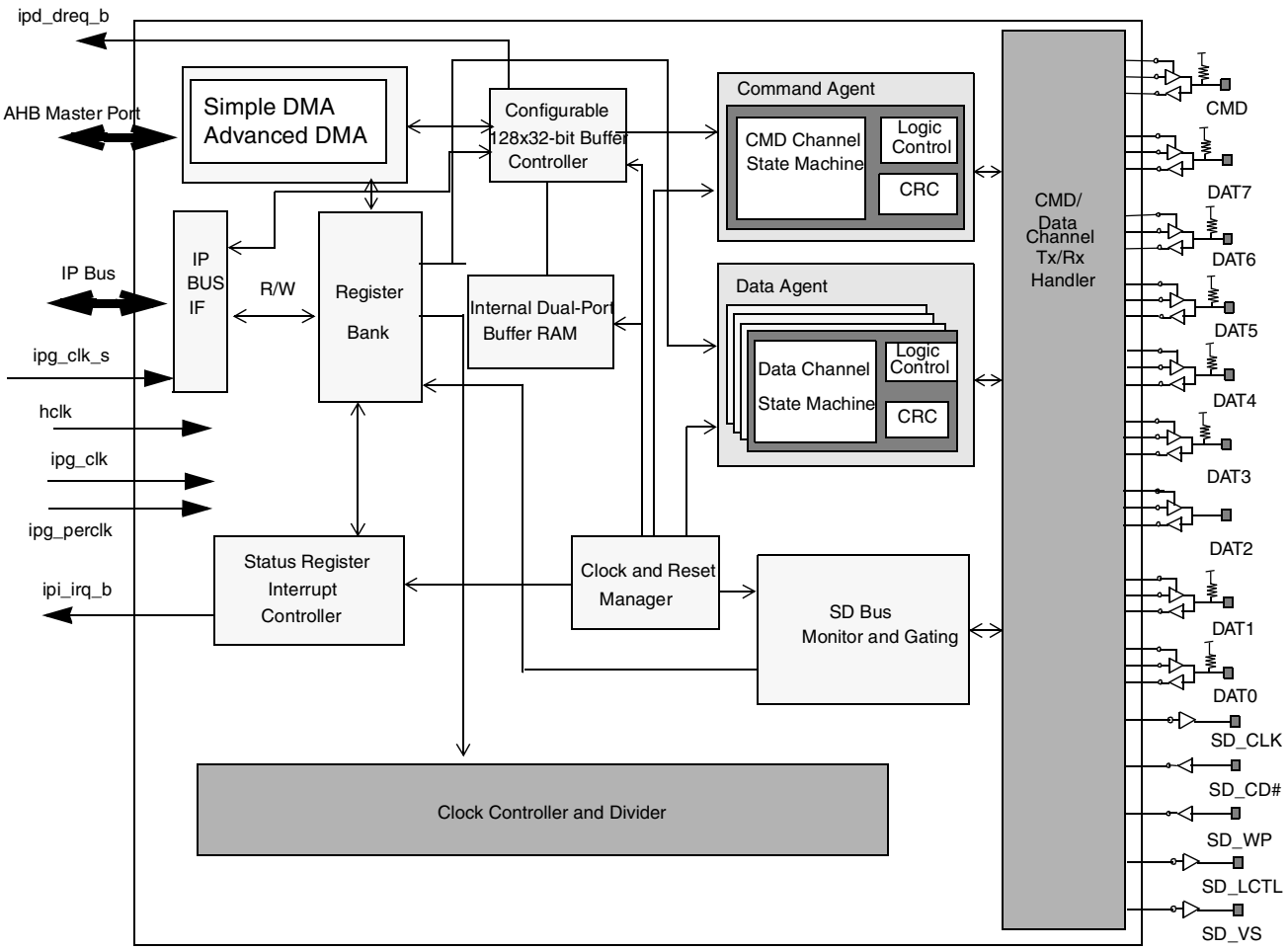


Figure 31-2. enhanced Secure Digital Host Controller (eSDHC) Block Diagram

31.2.2 Signal Descriptions

Table 31-1 shows properties of the I/O signals.

Table 31-1. Properties of I/O Signals

Name	Port	Function	Reset State	Pull up
SD_CLK	O	Clock for MMC/SD/SDIO card	0	N/A
SD_CMD	I/O	CMD line connected to card	1	Pull up
SD_DAT7	I/O	DAT7 line (MSB) in 8-bit mode Not used in other modes	1	Pull up
SD_DAT6	I/O	DAT6 line in 8-bit mode Not used in other modes	1	Pull up
SD_DAT5	I/O	DAT5 line in 8-bit mode Not used in other modes	1	Pull up

Table 31-1. Properties of I/O Signals (continued)

Name	Port	Function	Reset State	Pull up
SD_DAT4	I/O	DAT4 line in 8-bit mode Not used in other modes	1	Pull up
SD_DAT3	I/O	DAT3 line in 4/8-bit mode or configured as card detection pin May be configured as card detection pin in 1-bit mode	0	Pull-up/pull-down configurable
SD_DAT2	I/O	DAT2 line or read wait in 4-bit mode Read Wait in 1-bit mode	1	Pull up
SD_DAT1	I/O	DAT1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	1	Pull up
SD_DAT0	I/O	DAT0 (LSB) line in all modes Also used to detect busy state	1	Pull up
SD_CD#	I	Card detection signal If not used tie high	N/A	N/A
SD_WP	I	Card write protect detect If not used tie low	N/A	N/A
SD_OD	O	Open drain select (not generated within the eSDHC). Optional output	N/A	N/A
SD_LCTL	O	LED control used to drive an external LED Active high Fully controlled by the driver Optional output	0	N/A
SD_VS	O	Controls the voltage at SD_ pins. When asserted, voltage is high (around 3.0V); when negated, voltage is low (around 1.8V) Optional output	0	N/A

31.3 Memory Map and Register Definition

This section includes the module memory map and detailed descriptions of all registers.

31.3.1 Memory Map

Table 31-2 shows the eSDHC memory map. For the base address of a particular module instantiation, see the system memory map.

All registers support 32-bit accesses only.Addresses greater than 0x0044, except for 0x0050, 0x0054, 0x0058, and 0x00FC, are reserved and read as all zeros. Writes to these registers are ignored.

Table 31-2. eSDHC Memory Map [1]

Address (Register Abbreviation)	Register	Access	Section
0xBASE+0x000 (DSADDR)	DMA system address register	R/W	31.3.3.1/31-12
0xBASE+0x004 (BLKATTR)	Block attributes register	R/W	31.3.3.2/31-13
0xBASE+0x008 (CMDARG)	Command argument register	R/W	31.3.3.3/31-14
0xBASE+0x00C (XFERTYP)	Transfer type register	R/W	31.3.3.4/31-14
0xBASE+0x010 (CMDRSP0)	Command response 0 register	R	31.3.3.5/31-18
0xBASE+0x014 (CMDRSP1)	Command response 1 register	R	31.3.3.5/31-18
0xBASE+0x018 (CMDRSP2)	Command response 2 register	R	31.3.3.5/31-18
0xBASE+0x01C (CMDRSP3)	Command response 3 register	R	31.3.3.5/31-18
0xBASE+0x020 (DATPORT)	Data buffer access port register	R/W	31.3.3.6/31-20
0xBASE+0x024 (PRSSTAT)	Present state register	R	31.3.3.7/31-20
0xBASE+0x028 (PROCTL)	Protocol control register	R/W	31.3.3.8/31-25
0xBASE+0x02C (SYSCTL)	System control register	R/W	31.3.3.9/31-28
0xBASE+0x030 (IRQSTAT)	Interrupt status register	R	31.3.3.10/31-31
0xBASE+0x034 (IRQSTATEN)	Interrupt status enable register	R/W	31.3.3.11/31-36
0xBASE+0x038 (IRQSIGEN)	Interrupt signal enable register	R	31.3.3.12/31-38
0xBASE+0x03C (AUTOC12ERR)	Auto CMD12 status register	R	31.3.3.13/31-40
0xBASE+0x040 (HOSTCAPBL1T)	Host controller capabilities register	R	31.3.3.14/31-42
0xBASE+0x044 (WML)	Watermark level register	R/W	31.3.3.15/31-43
0xBASE+0x050 (FEVT)	Force event register	W	31.3.3.16/31-44
0xBASE+0x054 (ADMAES)	ADMA error status register	R	31.3.3.17/31-46
0xBASE+0x058 (ADSADDR)	ADMA system address register	R/W	31.3.3.18/31-48
0xBASE+0x0C0 (VENDOR)	Vendor-specific register	R/W	31.3.3.19/31-48
0xBASE+0x0FC (HOSTVER)	Host controller version register	R	31.3.3.20/31-49

[1] See Chapter 2, “Memory Map,” for the value of base address of eSDHC.

31.3.2 Register Summary

Table 31-3 summarizes the eSDHC registers.

Table 31-3. eSDHC Register Summary

Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (DSADDR)	R	DS_ADDR[31:16]															
	W																
	R	DS_ADDR[15:2]														0	0
	W																
0xBASE+0x004 (BLKATTR)	R	BLKCNT[15:0]															
	W																
	R	0	0	0	BLKSIZE[12:0]												
	W																
0xBASE+0x008 (CMDARG)	R	CMDARG[31:16]															
	W																
	R	CMDARG[15:0]															
	W																
0xBASE+0x00C (XFERTYP)	R	0	0	CMDINX[5:0]						CMDTYP [1:0]		DPS EL	CICE N	CCC EN	0	RSPTYP [1:0]		
	W																	
	R	0	0	0	0	0	0	0	0	0	0	MSB SEL	DTD SEL	0	AC12 EN	BCE N	DMA EN
	W																
0xBASE+0x010 (CMDRSP0)	R	CMDRSP0[31:16]															
	W																
	R	CMDRSP0[15:0]															
	W																
0xBASE+0x014 (CMDRSP1)	R	CMDRSP1[31:16]															
	W																
	R	CMDRSP1[15:0]															
	W																
0xBASE+0x018 (CMDRSP2)	R	CMDRSP2[31:16]															
	W																
	R	CMDRSP2[15:0]															
	W																

Table 31-3. eSDHC Register Summary (continued)

Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x01C (CMDRSP3)	R	CMDRSP3[31:16]															
	W																
	R	CMDRSP3[15:0]															
	W																
0xBASE+0x020 (DATPORT)	R	DATCONT[31:16]															
	W																
	R	DATCONT[15:0]															
	W																
0xBASE+0x024 (PRSSTAT)	R	DLSL[7:0]								CLSL	0	0	0	WPSPL	CDPL	0	CINS
	W																
	R	0	0	0	0	BREN	BWEN	RTA	WTA	SDOFF	PEROFF	HCKOFF	SDSTB	IPGOFF	DLA	CDIHB	CIHB
	W																
0xBASE+0x028 (PROCTL)	R	0	0	0	0	0	WECRM	WECINS	WECINT	0	0	0	0	IABG	RWCTL	CREQ	SABGREQ
	W																
	R	0	0	0	0	0	0	DMAS[1:0]		CDSS	CDTL	EMODE[1:0]		D3CD	DTW[1:0]		LCTL
	W																
0xBASE+0x02C (SYSCTL)	R	0	0	0	0	INITA	0	0	0	0	0	0	0	DTOCV[3:0]			
	W						RSTD	RSTC	RSTA								
	R	SDCLKFS[7:0]								DVS[3:0]				SDCLKEN	PEREN	HCKEN	IPGEN
	W																
0xBASE+0x030 (IRQSTAT)	R	0	0	0	DMAE	0	0	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
	W				w1c				w1c		w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	0	0	0	0	0	0	0	CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
	W								w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x034 (IRQSTATEN)	R	0	0	0	DMAESEN	0	0	0	AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
	W																
	R	0	0	0	0	0	0	0	CINTSEN	CRMSEN	CINSSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
	W																

Table 31-3. eSDHC Register Summary (continued)

Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x038 (IRQSIGEN)	R	0	0	0	DMAEIEN	0	0	0	AC12EIEN	0	DEBEIEN	DCEIEN	DTOEIEN	CIEIEN	CEBEIEN	CCEIEN	CTOEIEN
	W																
	R	0	0	0	0	0	0	0	CINTIEN	CRMIEN	CINSIEN	BRRIEN	BWRIEN	DINTIEN	BGEIEN	TCIEN	CCIEN
	W																
0xBASE+0x03C (AUTOC12ERR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	CNIBAC12E	0	0	AC12IE	AC12CE	AC12EBE	AC12TOE	AC12NE
	W																
0xBASE+0x040 (HOSTCAPBL1T)	R	0	0	0	0	0	VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MAXBL[1:0]		
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
0xBASE+0x044 (WML)	R	0	0	0	WR_BRST_LEN[3:0]					WR_WML[7:0]							
	W																
	R	0	0	0	RD_BRST_LEN[3:0]					RD_WML[7:0]							
	W																

Table 31-3. eSDHC Register Summary (continued)

Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x050 (FEVT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W	FEVTCINT			FEVTDMAE				FEVTAC12E		FEVTDEBE	FEVTDCE	FEVTDTOE	FEVTCIE	FEVTCEBE	FEVTCCE	FEVTCTOE
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W									FEVTCNIBAC12			FEVTAC12IE	FEVTAC12EBE	FEVTAC12CE	FEVTAC12TOE	FEVTAC12NE
0xBASE+0x054 (ADMAES)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	ADMADCE	ADMALME	ADMAES[1:0]	
	W																
0xBASE+0x058 (ADSADDR)	R W	ADS_ADDR[31:16]															
	R W	ADS_ADDR[15:0]															
0xBASE+0x0C0 (VENDOR)	R	0	0	0	0	DBG_SEL[3:0]				INT_ST_VAL[7:0]							
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VOLT_SEL	EXT_DMA_EN
	W																
0xBASE+0x0FC (HOSTVER)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	VVN[7:0]							SVN[7:0]								
	W																

31.3.3 Register Descriptions

This section provides detailed descriptions of the module's registers.

Figure 31-3 and Table 31-4 explain the conventions used in register diagrams and tables.

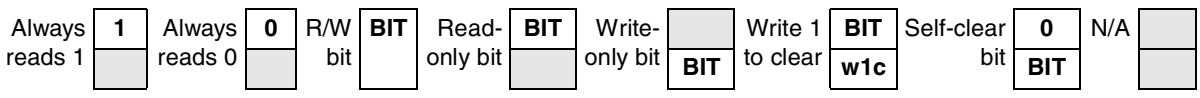


Figure 31-3. Register Field Conventions

Table 31-4. General Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can correspondingly be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that can be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to 0 (zero).
1	Resets to 1 (one).
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

NOTE

The term reset refers to power-on-reset (POR). The reset values given are POR reset values: these may differ from software reset values.

31.3.3.1 DMA System Address Register (DSADDR)

The DSADDR register contains the physical system memory address used for DMA transfers. Figure 31-4 shows the register. Table 31-5 describes the register fields.

Address 0xBASE+0x000 (DSADDR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DS_ADDR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DS_ADDR[15:2]														0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 31-4. DMA System Address Register Diagram (DSADDR)

Table 31-5. DMA System Address Register Field Descriptions

Field	Description
31–2 DS_ADDR[31:2]	DMA system address. This register contains the 32-bit system memory address for a DMA transfer. Since the address must be (4-byte) word-aligned, the last 2 bits are always read as 0. When the eSDHC stops a DMA transfer, this register points to the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value. The host driver initializes this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register. This register is protected during a data transfer. When data lines are active, writes to this register are ignored. The host driver waits until the DLA bit in the present state register is cleared before writing to this register. The eSDHC internal DMA only supports continuous physical memory access, and does not support a virtual memory system. Due to AHB burst limitations, if a burst of type SEQ crosses the 1 KB boundary, eSDHC I automatically changes the burst type to NSEQ. This register supports dynamic address reflection: when TC bit is set, it automatically alters the value of internal address counter. Software cannot change this register when TC bit is set.
1–0	Reserved, read as 0

31.3.3.2 Block Attributes Register (BLKATTR)

This register is used to configure the number of data blocks and the number of bytes in each block.

Figure 31-5 shows the register. Table 31-6 describes the register fields.

Address 0xBASE+0x004 (BLKATTR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BLKCNT[15:0][1]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	BLKSIZE[12:0]												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 31-5. Block Attributes Register (BLKATTR)

Table 31-6. Block Attributes Register Field Descriptions

Field	Description
31–16 BLKCNT	Block count for current transfer. This field is enabled when the block count enable (BCEN) bit in the transfer mode register is set to 1, and is valid only for multi-block transfers. For single block transfers, the register always reads as 1. For multi-block transfers, the host driver sets this register to a value between 1 and the maximum block count. The eSDHC decrements the block count after each block transfer, and stops when the count reaches zero. Setting the block count to 0 results in no data blocks being transferred. This register should be accessed only when no transaction is executing (that is, after transactions are stopped). Read operations on this register during data transfers can return an invalid value, and write operations are ignored. When saving transfer content as a result of a Suspend command, the number of blocks yet to be transferred can be determined by reading this register after transfer is paused by a stop at block gap operation, and before sending the Suspend command. After the Suspend command is sent the eSDHC regards the current transfer as aborted and changes the BLKCNT register back to its original value. When restoring transfer content prior to issuing a Resume command, the host driver is responsible to restore the previously-saved block count. 0x0000Stop count 0x0001 1 block 0x0002 2 blocks ..... 0xFFFF 65535 blocks **Note:** Although the BLKCNT field is 0 after reset, the read value after reset is 0x0001. This is because reset clears the MSBSEL bit to indicate a single-block transfer, so that BLKCNT reads as 0x0001.

Table 31-6. Block Attributes Register Field Descriptions (continued)

Field	Description
15–13	Reserved, read as 0
12–0 BLKSIZE[12:0]	Transfer block size. This register specifies the block size (in bytes) for block data transfers. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0x000) No data transfer 0x001) 1 Byte 0x002) 2 Bytes … 0x1000) 4096 Bytes 0x1001–0xFFFF Reserved

31.3.3.3 Command Argument Register (CMDARG)

Figure 31-6 shows the register. Table 31-7 describes the register fields.

Address 0xBASE+0x008 (CMDARG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CMDARG[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMDARG[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-6. Command Argument Register (CMDARG)

Table 31-7. Command Argument Register Field Descriptions

Field	Description
31–0 CMDARG[31:0]	Command argument. The SD/MMC command argument is specified as bits 39–8 of the command format in the SD and MMC specifications.This register is write-protected when the CIHB bit is set in the present state register.

31.3.3.4 Transfer Type Register (XFERTYP)

The XFERTFYP register is used to control the operation of data transfers. The host driver is responsible to set this register before issuing a command followed by a data transfer, or before issuing a Resume command. During data transfers, to prevent data loss the eSDHC prevents writing to the following bits: DPSEL, MBSEL, DTDSEL, AC12EN, BCEN and DMAEN.

The host driver is responsible to check the command inhibit bits (CDIHB and CIHB) in the present state register before writing to this register. When the CDIHB bit is set, any attempt to send a command with data transfer by writing to this register is ignored; when the CIHB bit is set, any write to this register is ignored.

When sending commands followed by data transfer, the command is ignored unless the following conditions are met:

- Block size is nonzero (specified by the BLKSIZE field in the BLKATTR register).
- At least one of the following conditions holds:
  - Block count is nonzero (specified by the BLKCNT field in the BLKATTR register).
  - Single-block transfer is indicated (MSBSEL bit in XFERTYP register is cleared)
  - Block count is disabled (BCEN bit in XFERTYP register is cleared)
- (Write commands only) Write protect switch is inactive (WPSPL bit in present state register is set to 1).

After the command followed by data transfer is sent, if no response is received within 64 clock cycles (the response timeout period) then the eSDHC aborts the data transfer. Response timeout occurs for instance in the following cases:

- The card responds to the command, but eSDHC does not receive the response
- An internal DMA (either simple DMA or ADMA) read operation occurs, so that external system memory is overwritten by the internal DMA with data sent back from the card.

If the command times out, the driver is responsible to reissue the command.

Figure 31-7 shows the register. Table 31-8 describes the register fields.

Address 0xBASE+0x00C (XFERTYP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CMDINX[5:0]						CMDTYP [1:0]		DPSEL	CICEN	CCCEN		RSPTYP [1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	MSBSEL	DTDSEL	0	AC12EN	BCEN	DMAEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 31-7. Transfer Type Register (XFERTYP)

Table 31-8. Transfer Type Register Field Descriptions

Field	Description
31–30	Reserved, read as 0
29–24 CMDINX[5:0]	Command index. These bits are set to the command number that is specified in bits 45-40 of the command format in the SD Memory Card Physical Layer Specification and the SDIO Card Specification.
23–22 CMDTYP[1:0]	Command type. There are three types of special commands: Suspend, Resume and Abort (for all other commands, CMDTYP is set to 0b00). The three special commands are described as follows: • Suspend command: If a Suspend command is sent, the eSDHC assumes that the card bus has been released and that it can issue the next command which uses the DAT line. However, the eSDHC does not monitor the content of command response, so it does not know if the Suspend command succeeds or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform the eSDHC that a Suspend command was successfully issued. Refer to Section 31.5.3.3.1, Suspend Operation" for more details. After the end bit of the command is sent, the eSDHC negates read wait for read transactions and stops checking busy for write transactions. In 4-bit mode, the interrupt cycle starts. If the Suspend command fails, the eSDHC maintains its current state, and the host driver restarts the transfer by setting the continue request bit in the protocol control register. • Resume command: The host driver restarts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The eSDHC checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, the eSDHC stops reads to the buffer. If this command is set when executing a write transfer, the eSDHC stops driving the DAT line. After issuing the Abort command, the host driver should issue a software reset (Abort transaction). 00 Normal— all other commands 01 Suspend CMD52 for writing bus suspend in CCCR 10 Resume CMD52 for writing function select in CCCR 11 Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select. This bit is set to 1 to indicate that data is ready to be transferred using the DAT line. It is set to 0 in the following cases: • Commands using only the CMD line (such as CMD52). • Commands that transfer no data but use the busy signal on DAT[0] line (R1b or R5b, such as CMD38) 0 No data present 1 Data present **Note:** • The Resume command requires this bit be set to 1, while the other XFERTYP fields (except for CMDTYP) are the same as when the transfer was initially launched. • When the write protect switch is on, (WPSPL bit is cleared in the present state register), all writes to the transfer type register with DTDSEL = 0 and DPSEL = 1 are ignored.
20 CICEN	Command index check enable. When this bit is set to 1, the eSDHC checks the index field in the response to see if it has the same value as the command index. If not, the eSDHC reports a command index error. The Index field is not checked if the bit is cleared. 0 Disable command index check 1 Enable command index check
19 CCCEN	Command CRC check enable. If this bit is set to 1, the eSDHC checks the CRC field in the response. If an error is detected, the eSDHC reports a command CRC error. The CRC field is not checked if this bit is cleared. The number of bits checked by the CRC field value changes according to the length of the response. (Refer to RSPTYP[1:0] and Table 31-10.) 0 Disable command CRC check 1 Enable command CRC check

Table 31-8. Transfer Type Register Field Descriptions (continued)

Field	Description
18	Reserved
17–16 RSPTYP[1:0]	Response type select: 00 No response 01 Response length 136 10 Response length 48 11 Response length 48, check busy after response
15–6	Reserved, read as 0
5 MSBSEL	Multi / single-block select: This bit enables multi-block DAT line data transfers. For any other commands, this bit is set to 0. If this bit is 0, it is not necessary to set the block count register. Table 31-9 shows MSBSEL settings corresponding to different transfer types. 0 Single-block transfer 1 Multi-block transfer
4 DTDSEL	Data transfer direction select. This bit defines the direction of DAT line data transfers. The host driver sets this bit to 1 to transfer data from the SD card to the eSDHC. This bit is cleared for all other commands. 0 Write (host to card) 1 Read (card to host)
3	Reserved, read as 0
2 AC12EN	Auto CMD12 enable. Multi-block transfers for memory require a CMD12 to stop the transaction. When this bit is set to 1, the eSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver does not set this bit to issue commands that do not require CMD12 to stop a multi-block data transfer. In particular, secure commands defined in the file security specification do not require CMD12. In single-block transfers, the eSDHC ignores this bit. 0 Disable auto CMD12 1 Enable auto CMD12
1 BCEN	Block count enable. This bit enables the block count register, which is only relevant for multi-block transfers. When this bit is cleared, the internal block counter is disabled: this setting is used for infinite transfers.Table 31-9 shows BCEN settings corresponding to different transfer types. 0 Disable block count register 1 Enable block count register
0 DMAEN	DMA enable. This bit enables internal DMA functionality. If this bit is set to 1, an internal DMA operation begins when the host driver sets the DPSEL bit of this register. Whether the simple DMA, or the advanced DMA, is active depends on the DMA select field of the protocol control register. 0 Disable DMA 1 Enable DMA

Table 31-9 shows multi/single-block select (MSBSEL) and block count enable (BCEN) settings for different data transfer types.

Table 31-9. MSBSEL and BCEN Bit Settings for Different Transfer Types

Multi/Single Block Select	Block Count Enable	Block Count	Function
0	Don’t Care	Don’t Care	Single Transfer
1	0	Don’t Care	Infinite Transfer

Table 31-9. MSBSEL and BCEN Bit Settings for Different Transfer Types (continued)

Multi/Single Block Select	Block Count Enable	Block Count	Function
1	1	Positive Number	Multiple Transfer
1	1	Zero	No Data Transfer

Table 31-10 shows the settings for responses type select (RSPTYP), index check enable (CICEN), and command CRC check enable (CCCEN) fields in the XFERTYP register corresponding to different response types.

Table 31-10. RSPTYP, CICEN, and CCCEN Settings for Different Response Types

Response Type Select (RSPTYP)	Index Check Enable (CICEN)	CRC Check Enable (CCCEN)	Response Type
00	0	0	No response
01	0	1	R2
10	0	0[1]	R3,R4
10	1	1	R1,R5,R6
11	1	1	R1b,R5b[2]

[1] The CRC field for R3 and R4 is expected to be all 1's. The CRC check is disabled for these response types.

[2] The SDIO Specification does not distinguish between R5 and R5b. The notation R5b here indicates the case where eSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.

31.3.3.5 Command Response *n* Register (CMDRSP0–CMDRSP3)

The CMDRSP*n* registers are used to store parts *n* of the response bits from the card (*n* = 0,1,2,3).

Figure 31-8 shows the register. Table 31-11 describes the register fields.

Address 0xBASE+0x010 (CMDRSP0)
0xBASE+0x014 (CMDRSP1)
0xBASE+0x018 (CMDRSP2)
0xBASE+0x01C (CMDRSP3)

Access: User read

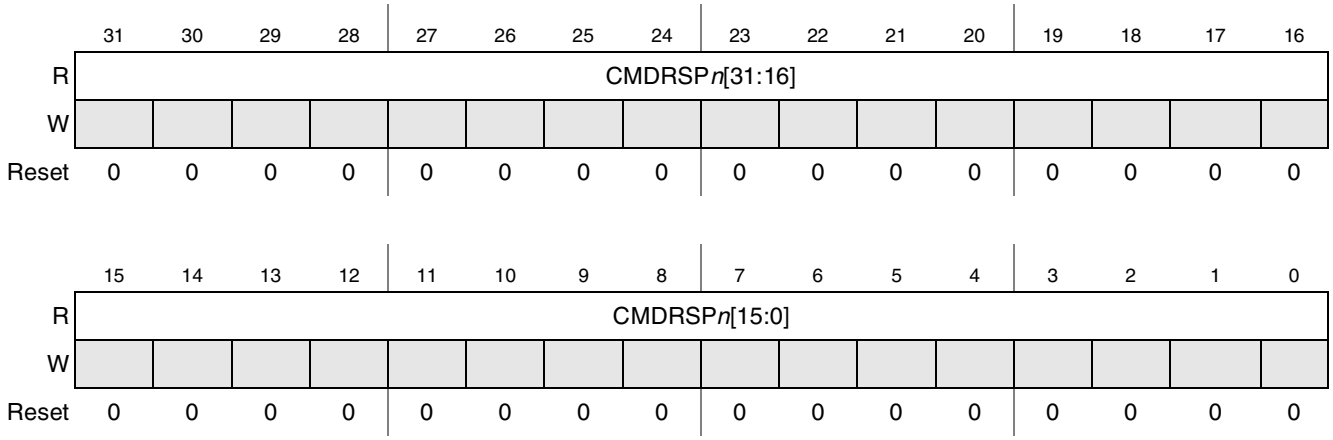


Figure 31-8. Command Response *n* Register (CMDRSP0-CMDRSP3)

Table 31-11. Command Response *n* Register Field Description

Field	Description
31–0 CMDRSP*n*[31:0]	Command response *n*. Refer to Table 31-12 for the mapping of command responses from the SD bus to this register for each response type.

Table 31-12 describes the mapping of command responses from the SD bus to command response registers for each response type. In the table, R[*x*:*y*] refers to a bit range within the response data as transmitted on the SD bus.

Table 31-12. Response Field Definitions for Each Response Type

Response Type	Meaning of Response	Response Field	Response Register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O and others	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (Publish RCA)	New Published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

The eSDHC only stores part of the response data in the command response registers. This enables the host driver to efficiently read 32 bits of response data in one read cycle on a 32-bit bus system.The eSDHC checks the response data's index field and the CRC (as specified by the command index check enable and the command CRC check enable bits in the transfer type register) and generates an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, the eSDHC checks R[47:1], and if the response length is 136 the eSDHC checks R[119:1].

Table 31-12 shows that the auto CMD12 and CMD_wo_DAT responses are stored in the CMDRSP3 and CMDRSP0 registers, respectively. This prevents overwriting of the auto CMD12 response with the CMD_wo_DAT (or vice versa) when the corresponding commands are executed concurrently. When the eSDHC modifies part of the command response registers as shown in Table 31-12, it preserves the unmodified bits.

31.3.3.6 Buffer Data Port Register (DATPORT)

This register contains the buffer data port. Figure 31-9 shows the register. Table 31-13 describes the register fields. The 32-bit DATPORT register is used for CPU or external DMA access the internal buffer. When the internal DMA is enabled, writes to this register is ignored, and the register is read as 0.

Address 0xBASE+0x020 (DATPORT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATCONT[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATCONT[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-9. Buffer Data Port Register (DATPORT)

Table 31-13. Buffer Data Port Register Field Descriptions

Field	Description
31–0 DATCONT[31:0]	Data content. When the internal DMA is enabled, writes to this register are ignored, and the register is read as 0.

31.3.3.7 Present State Register (PRSSTAT)

The 32-bit read-only PRSSTAT register provides the host driver with the eSDHC status.

Figure 31-10 shows the register. Table 31-14 describes the register fields.

Address 0xBASE+0x024 (PRSSTAT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLSL[7:0]								CLSL	0	0	0	WPSPL	CDPL	0	CINS
W																
Reset	1	1	1	1	*n*[1]	1	1	1	1	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	BREN	BWEN	RTA	WTA	SDOFF	PEROFF	HCKOFF	IPGOFF	SDSTB	DLA	CDIHB	CIHB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1 The reset value of DLSL [3] depends on the pull-up/pull-down configuration of DAT[3] (see Table 31-1). If DAT[3] is pulled down, then DLSL[3] resets to 0; if DAT[3] is pulled up, then DLSL[3] resets to 1.

Figure 31-10. Present State Register (PRSSTAT)

Table 31-14. Present State Register Field Descriptions

Field	Description
31–24 DLSL[7:0]	DAT[7:0] line signal level. These status bits are used to check the DAT line levels to recover from errors, and for debugging.DLSL[0] is especially useful for detecting the busy signal level. The reset value is affected by the external pull-up / pull-down resistors. These bits all reset to 1 except for DLSL[3], whose reset value depends on the pull-up / pull-down configuration of DAT[3]. If DAT[3] is pulled down, then DLSL[3] resets to 0; if DAT[3] is pulled up, then DLSL[3] resets to 1. DAT[*n*]: Data *n* line signal level (*n* = 7…0)
23 CLSL	CMD line signal level. This status bit is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up/pull-down resistor—by default, the read value of this bit after reset is 0b1.
22–20	Reserved, read as 0
19 WPSPL	Write protect switch pin level. The write protect switch is supported for memory and combo cards.This bit reflects the inverted value of the SD_WP pin of the card socket. A software reset does not affect this bit. The reset value is effected by the external write protect switch. If the SD_WP pin is not used, it should be tied low, so that the reset value of this bit is high and write is enabled. 0 Write protected (SD_WP=1) 1 Write enabled (SD_WP=0)

Table 31-14. Present State Register Field Descriptions (continued)

Field	Description
18 CDPL	Card detect pin level. This bit reflects the inverse value of the SD_CD# signal for the card socket: for instance, when a card is inserted in the socket the SD_CD# input signal is negated, and the CDPL bit is set to 1. Debouncing is not performed on this bit. This bit may be valid, but is not guaranteed because of propagation delay. Use of this bit is limited to testing since it must be debounced by software. A software reset does not effect this bit. A write to the force event register does not effect this bit. The reset value is effected by the external card detection pin. 0 No card present (SD_CD# = 1) 1 Card present (SD_CD# = 0)
17	Reserved, read as 0
16 CINS	Card inserted. This bit indicates whether a card has been inserted. The eSDHC debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a card insertion interrupt in the interrupt status register. Changing from a 1 to 0 generates a card removal interrupt in the interrupt status register. A write to the force event register does not effect this bit. Setting the software reset for all (RSTA) bit in the system control register does not effect this bit. A software reset does not effect this bit. 0 Power on reset or no card 1 Card inserted
15–12	Reserved, read as 0
11 BREN	Buffer read enable. This read-only flag indicates that valid data exists in the host-side buffer.When this bit is set to 1, valid data in the buffer meets or exceeds the read watermark level. This bit changes from 1 to 0 whenever data is read from the buffer. This bit changes from 0 to1 when valid data in the buffer meets or exceeds the watermark level, and the buffer read ready interrupt has been generated and enabled. 0 Read disable 1 Read enable This status bit is used for non-DMA read transfers. The eSDHC may implement multiple buffers to transfer data efficiently.
10 BWEN	Buffer write enable. This read-only flag indicates that space is available for write data. If this bit is set to 1, available space in the buffer meets or exceeds the write watermark level]. This bit changes from 1 to 0 when data is written to the buffer. This bit changes from 0 to 1 when the available space in the buffer meets or exceeds the write watermark level, and the buffer write ready interrupt is generated and enabled. 0 Write disable 1 Write enable This status bit is used for non-DMA write transfers. The eSDHC can implement multiple buffers to transfer data efficiently.
9 RTA	Read transfer active. This status bit is used to detect completion of a read transfer. This bit is set in either of the following two cases: • After the end bit of the read command. • When the continue request bit in the protocol control register is set to 1 to restart a read transfer. A transfer complete interrupt is generated when the RTA bit is cleared. The bit is cleared in either of the following two cases: • When the last data block as specified by the block size is transferred to the system, so that all data are read from the eSDHC internal buffer. • When all valid data blocks have been transferred from eSDHC internal buffer to the system and no current block transfers are being sent because the stop at block gap request is set to 1. 0 No valid data 1 Transferring data

Table 31-14. Present State Register Field Descriptions (continued)

Field	Description
8 WTA	Write transfer active. This status bit indicates a write transfer is active. If this bit is cleared, it means no valid write data exists in the eSDHC. This bit is set in either of the following cases: • After the end bit of the write command. • When the continue request bit in the protocol control register is set to 1 to restart a write transfer. This bit is cleared in either of the following cases: • After receiving the CRC status of the last data block as specified by the transfer count (single-block or multi-block). • After receiving the CRC status of the current block, when data transmission is about to be stopped by a stop at block gap request. A block gap event interrupt is generated when this bit is cleared as result of a stop at block gap request during a write transaction. The host driver can use this status bit to determine when to issue commands during write busy state. 0 No valid data 1 Transferring data
7 SDOFF	SD clock gated off internally. This status bit indicates that the SD clock is internally gated off due to one of the following causes: • Buffer overrun or underrun • Read pause without read wait assertion • The driver has cleared the SDCLKEN bit to stop the SD clock. The host driver can use this bit to debug data transactions on the SD bus. 0 SD clock is active. 1 SD clock is gated off.
6 PEROFF	Peripheral source clock (ipg_perclk) gated off internally. This status bit indicates that the ipg_perclk is internally gated off. The host driver can use this bit for debug purposes during transactions on the SD bus. 0 ipg_perclk is active. 1 ipg_perclk is gated off.
5 HCKOFF	Master clock (hclk) gated off internally. This status bit indicates that the hclk is internally gated off. The host driver can use this bit for debug purposes during data transfers. 0 hclk is active. 1 hclk is gated off.
4 IPGOFF	ipg_clk gated off internally. This status bit indicates that the ipg_clk is internally gated off. The host driver can use this bit for debug purposes. 0 ipg_clk is active. 1 ipg_clk is gated off.
3 SDSTB	SD clock stable. This status bit indicates that the internal card clock is stable. This bit is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear SDCLKEN bit in system control register to remove glitches on the card clock when the frequency is changing. 0 Clock is changing frequency and not stable. 1 Clock is stable.

Table 31-14. Present State Register Field Descriptions (continued)

Field	Description
2 DLA	Data line active. This status bit indicates whether one of the DAT lines on the SD bus is in use. In the case of read transactions: This bit indicates if a read transfer is executing on the SD bus. A change in this bit from 1 to 0 between data blocks generates a block gap event interrupt in the interrupt status register. This bit is set in either of the following cases: • After the end bit of the read command. • When writing a 1 to the continue request bit in the protocol control register to restart a read transfer. This bit is cleared in either of the following cases: • When the end bit of the last data block is sent from the SD bus to the eSDHC. • When the read wait state is stopped by a suspend command and the DAT2 line is released. The eSDHC waits at the next block gap by driving a read wait signal at the start of the interrupt cycle. If the read wait signal is already driven (indicating that the data buffer cannot receive data), the eSDHC can wait for a current block gap by continuing to drive the read wait signal. It is necessary to support read wait in order to use the suspend / resume function. This bit remains set to 1 during read wait. In the case of write transactions: This bit indicates that a write transfer is executing on the SD bus. Changes in this bit from 1 to 0 generate a transfer complete interrupt in the interrupt status register. This bit is set in either of the following cases: • After the end bit of the write command. • When writing to 1 to the continue request bit in the protocol control register to continue a write transfer. This bit is cleared in either of the following cases: • When the SD card releases write busy of the last data block, the eSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, the eSDHC assumes the card drives the not busy signal. • When the SD card releases write busy, prior to waiting for write transfer, and as a result of a stop at block gap request. In the case of command with busy pending: This status bit indicates that a busy state follows the command and the data line is in use. This bit is cleared when the DAT0 line is released. 0 DAT line inactive 1 DAT line active

Table 31-14. Present State Register Field Descriptions (continued)

Field	Description
1 CDIHB	Command inhibit (DAT). This status bit is generated if either the DAT line active (DLA) bit or the read transfer active (RTA) bit in this register is set to 1. If this bit is cleared, it indicates that the eSDHC can issue the next SD/MMC command. Commands with a busy signal (for example, R1b, R5b type) belong to command inhibit (DAT). Except in the case when the command busy is finished, changing this bit from 1 to 0 generates a transfer complete interrupt in the interrupt status register. **Note:** The SD host driver can save registers for a suspend transaction after this bit has changed from 1 to 0. 0 Can issue command which uses the DAT line 1 Cannot issue command which uses the DAT line
0 CIHB	Command inhibit (CMD). This status bit is cleared when the CMD line is not in use, indicating that the eSDHC can issue a SD/MMC command using the CMD line. This bit is set immediately after the transfer type register is written. This bit is cleared when the command response is received. Changing this bit from 1 to 0 sets the command complete (CC) bit in the interrupt status register, which generates an interrupt if the CCIEN bit is set in the interrupt signal enable register. Commands using only the CMD line can be issued if this bit is cleared, even when CDIHB (bit 1 of this register) is set. Examples of commands using only the CMD line include CMD0, CMD12, CMD13 (for memory cards) and CMD52 (for SDIO). This bit is remains set (and the CC bit in the interrupt status register remains cleared) in either of the following two cases: • If the eSDHC detects a CMD line conflict when the command is issued (eSDHC drives the CMD line to 1, but detects a 0 at the next SD_CLK edge). In this case, command CRC error and command timeout error bits in the interrupt status register are set to 1. • If the command is not issued due to an auto CMD12 error. 0 Can issue command using only the CMD line 1 Cannot issue command

31.3.3.8 Protocol Control Register (PROCTL)

Figure 31-11 shows the register. Table 31-15 describes the register fields.

Address 0xBASE+0x028 (PROCTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	WECRM	WECINS	WECINT					IABG	RWCTL	CREQ	SABGREQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DMAS[1:0]		CDSS	CDTL	EMODE[1:0]		D3CD	DTW[1:0]		LCTL
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 31-11. Protocol Control Register (PROCTL)

Table 31-15. Protocol Control Register Field Descriptions

Field	Description
31–27	Reserved, read as 0
26 WECRM	Wakeup event enable on SD card removal. This bit enables a wakeup event (via a card removal) in the interrupt status register. FN_WUS (wakeup support) in CIS does not effect this bit. When this bit is set, the card removal status and the eSDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the card removal status and assert the eSDHC interrupt. 0 Disable 1 Enable
25 WECINS	Wakeup event enable on SD card insertion. This bit enables a wakeup event, via a card insertion, in the interrupt status register. FN_WUS (wakeup support) in CIS does not effect this bit. When this bit is set, the card insertion status and the eSDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the card insertion status and assert the eSDHC interrupt. 0 Disable 1 Enable
24 WECINT	Wakeup event enable on card interrupt. This bit enables a wakeup event, via a card interrupt, in the interrupt status register. This bit can be set to 1 if FN_WUS (wakeup support) in CIS is set to 1. When this bit is set, the card interrupt status and the eSDHC interrupt can be asserted without SD_CLK toggling. When the wakeup feature is not enabled, the SD_CLK must be active in order to assert the card interrupt status and assert the eSDHC interrupt. 0 Disable 1 Enable
23–20	Reserved
19 IABG	Interrupt at block gap. This bit is only valid for SDIO cards in 4-bit mode. It selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multi-block transfer. Clearing the bit disables interrupt detection during a multi-block transfer. If the SDIO card cannot signal an interrupt during a multi-block transfer, this bit should be cleared to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it sets this bit according to the CCCR of the card. 0 Disabled 1 Enabled
18 RWCTL	Read wait control. The read wait function is optional for SDIO cards. If the card supports read wait, setting this bit enables the read wait protocol to stop read data using the DAT[2] line. Otherwise the eSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it sets this bit according to the CCCR of the card. If the card does not support read wait, this bit is never set to 1—otherwise DAT line conflicts may occur. If this bit is cleared and the stop at block gap request (SABGREQ) bit is set, the eSDHC stops the SD clock to pause the read operation. 0 Disable read wait control, and stop SD Clock at block gap when SABGREQ bit is set 1 Enable read wait control, and assert read wait without stopping SD clock at block gap when SABGREQ bit is set
17 CREQ	Continue request. When a Suspend operation is not accepted by the card, setting this bit restarts the paused transfer. This bit is also used to restart a transaction that was stopped using the stop at block gap request (SABGREQ). To cancel the stop at block gap, the host driver clears the SABGREQ bit and sets this bit to 1 to restart the transfer. If both the SABGREQ bit and this bit are set to 1, the continue request is ignored. The eSDHC automatically clears this bit: the host driver does not need to clear it. 0 No effect 1 Restart

Table 31-15. Protocol Control Register Field Descriptions (continued)

Field	Description
16 SABGREQ	Stop at block gap request. This bit is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. In the case of read transfers, setting the SABREQ bit stops the transaction at the next block gap only if the SDIO card supports read wait, and the read wait control bit (RWCTL) in this register is set to 1. Otherwise, the read operation can be paused at the next block gap by stopping the SD bus clock. In the case of write transfers in which the host driver writes data to the data port register, the host driver sets this bit after all block data is written. If this bit is set to 1, the host driver does not write data to the data port register after a block is sent. If the host driver clears this bit before the transfer complete bit in interrupt status register is set:, then the eSDHC's behavior is unpredictable. Clearing both SABGREQ and CREQ bits does not cause the transaction to restart. This bit affects the read transfer active, write transfer active, DAT line active and command inhibit (DAT) bits in the present state register. 0 Transfer 1 Stop
15–10	Reserved, read as 0
9–8 DMAS	DMA select. This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation. 00 No DMA or simple DMA is selected 01 ADMA1 is selected 10 ADMA2 is selected 11 Reserved
7 CDSS	Card detect signal selection. This bit selects the source for the card detection. 0 Card detection level is selected (for normal purposes) 1 Card detection test level is selected (for test purposes)
6 CDTL	Card detect test level. This is bit is enabled while the card detection signal selection is set to 1 and it indicates card insertion. 0 Card detect test level is 0, no card inserted 1 Card detect test level is 1, card inserted
5–4 EMODE	Endian mode. The eSDHC supports all four endian modes in data transfer. Refer to Section 31.4.1, Data Buffer" for more details. 00 Big-endian mode 01 Halfword big-endian mode 10 Little-endian mode 11 Reserved
3 D3CD	DAT3 as card detection pin. If this bit is set, DAT3 should be pulled down to act as a card detection pin. 0 DAT3 does not monitor card insertion 1 DAT3 as card detection pin **Note:** Since DAT3 is also a chip select for SPI mode, a pull-down on this pin and CMD0 may set the card into the SPI mode. eSDHC does not support SPI mode.

Table 31-15. Protocol Control Register Field Descriptions (continued)

Field	Description
2–1 DTW[1:0]	Data transfer width. This bit selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00 1-bit mode 01 4-bit mode 10 8-bit mode 11 Reserved
0 LCTL	LED control. This bit, fully controlled by the host driver, is used to caution the user not to remove the card while the card is being accessed. When the software issues multiple SD commands, setting the bit once before the first command is sufficient: it is not necessary to reset the bit between commands. 0 LED off 1 LED on

There are three ways to restart the transfer after stop at the block gap, depending on the status of the Suspend command.

- If the host driver does not issue a Suspend command, the continue request is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the continue request is used to restart the transfer.

When a stop at block gap request stops the data transfer, the host driver waits for the transfer complete bit (interrupt status register) to be set before attempting to restart the transfer. When restarting the data transfer by continue request, the host driver clears the stop at block gap request before, or simultaneously with, issuing the continue request.

31.3.3.9 System Control Register (SYSCTL)

Figure 31-12 shows the register. Table 31-16 describes the register fields.

Address 0xBASE+0x02C (SYSCTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	INITA	0	0	0	0	0	0	0	DTOCV[3:0]			
W						RSTD	RSTC	RSTA								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDCLKFS[7:0]								DVS[3:0]				SDCLKEN	PEREN	HCKEN	IPGEN
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 31-12. System Control Register (SYSCTL)

Table 31-16. System Control Register Field Descriptions

Field	Description
31–28	Reserved, read as 0
27 INITA	Initialization active. When this bit is set, 80 SD clocks are sent to the card. After the 80 clocks are sent, this bit self clears. This bit is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this bit is already 1 has no effect. Writing 0 to this bit at any time has no effect. When either of the CIHB and CDIHB bits in the present state register are set, writes to this bit is ignored (when the command or data lines are active, writes to this bit is not allowed). However, when this bit is set to 1 (that is, during the initialization active period), it is still possible to issue a command—the command bit stream appears on the CMD signal after the 80 clock cycles are completed. In this way, the driver can ensure that 80 clock cycles have completed before the command is sent out. This is a useful feature in the case where the driver needs to send 80 cycles to the card and does not want to wait until this bit is self-cleared.
26 RSTD	Software reset for DAT line. Resets part of the data circuit and the DMA circuit. Setting this bit clears and initializes the buffer, and also clears the following bits: • Data port register: all bits • Present state register: buffer read enable, buffer write enable, read transfer active, write transfer active, data line active, command inhibit (DAT) • Protocol control register: Continue request, Stop at block gap request • Interrupt status register: buffer read ready, buffer write ready, DMA interrupt, block gap event, transfer complete 0 No reset 1 Reset
25 RSTC	Software reset for CMD line. Resets part of the command circuit. The following bits are cleared by this bit: • Command inhibit (CMD) bit in present state register • Command complete (CC) bit in interrupt status register 0 No reset 1 Reset
24 RSTA	Software reset for all. This reset effects the entire host controller except for the card detection circuit. Register bits of type ROC, RW, RW1C, RWAC are cleared. During initialization, the host driver sets this bit to 1 to reset the eSDHC. The eSDHC resets this bit to 0 when the capabilities registers are valid and the host driver can read them. Additional use of software reset for all does not affect the value of the capabilities registers. After this bit is set, it is recommended that the host driver reset and reinitialize the external card. 0 No Reset 1 Reset
23–20	Reserved, read as 0

Table 31-16. System Control Register Field Descriptions (continued)

Field	Description
19–16 DTOCV	Data timeout counter value. This value determines the interval by which DAT line timeouts are detected. Refer to the data timeout error bit in the interrupt status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SD_CLK frequency by this value. The host driver can clear the data timeout error status enable (in the interrupt status enable register) to prevent inadvertent time-out events. 0000 SD_CLK x 2^{13} 0001 SD_CLK x 2^{14} ..... 1110 SD_CLK x 2^{27} 1111 Reserved
15–8 SDCLKFS	SD clock (SD_CLK) frequency prescale select. This register is used to select the prescaler of the SD_CLK signal. The frequency of the SD clock is set based on SD_CLKF and DVS settings according to the following formula: SD clock frequency = (peripheral source clock frequency) / (SDCLKF x DVS) See Section 31.4.6, SD Clock Generator" for more information about SD clock generation. Multiple bits in this field must not be set, or the behavior of the prescaler is undefined. 0x00 Base clock (10–63 MHz) divided by 1 0x01 Base clock divided by 2 0x02 Base clock divided by 4 0x04 Base clock divided by 8 0x08 Base clock divided by 16 0x10 Base clock divided by 32 0x20 Base clock divided by 64 0x40 Base clock divided by 128 0x80 Base clock divided by 256 (reset value) Other settings Reserved, not allowed
7–4 DVS[3:0]	Frequency divisor. The frequency of the SD clock is set based on SDCLKF and DVS settings according to the following formula: SD clock frequency = (peripheral source clock frequency) / (SDCLKF x DVS) DVS provides a finer granularity for the available frequencies. Odd divisors are supported without deterioration of the duty cycle. See Section 31.4.6, SD Clock Generator" for more information about SD clock generation. 0x0 Divide by 1 0x1 Divide by 2 ... 0xE Divide by 15 0xF Divide by 16
3 SDCLKEN	SD clock (SD_CLK) enable. The host controller stops SD_CLK when this bit is cleared. The SD_CLK frequency can only be changed when this bit is cleared. If the card inserted bit in the present state register is cleared, the host driver saves power by clearing this bit. 0 SD clock is stopped 1 SD clock is enabled

Table 31-16. System Control Register Field Descriptions (continued)

Field	Description
2 PEREN	Peripheral source clock (ipg_perclk) enable. When this bit is set, ipg_perclk is always active and no automatic gating is applied. In this case SD_CLK is active except during auto gating-off in case of buffer danger (pending underrun or overrun).When this bit is cleared, the ipg_perclk is automatically gated off when there is no transaction on the SD bus, and when none of the following conditions are met: • The cmd part is reset • Data part is reset • A soft reset • The cmd is about to be sent • Clock divisor is just updated • Continue request is just set • Card insertion is detected • Card removal is detected • Card external interrupt is detected • 80 clocks for initialization phase is ongoing Since this bit is only a feature-enabling bit, clearing this bit does not stop SD_CLK immediately. 0 ipg_perclk is internally gated off. 1 ipg_perclk is not automatically gated off and is active.
1 HCKEN	Master clock (hclk) enable. If this bit is set, hclk is always active and no automatic gating is applied. When this bit is cleared, hclk is automatically off when no data transfer is active on the SD bus. 0 hclk is internally gated off 1 hclk is not automatically gated off
0 IPGEN	ipg_clk enable. If this bit is set, ipg_clk is always active and no automatic gating is applied. When this bit is cleared, the ipg_clk is internally gated off if none of the following conditions is met: • The cmd part is reset • Data part is reset • Soft reset • The cmd is about to be sent • Clock divisor is just updated • Continue request is just set • Card insertion is detected • Card removal is detected • Card external interrupt is detected • The ipg_perclk is not gated off (thus clearing this bit has no effect unless the PEREN bit is also cleared) 0 ipg_clk is internally gated off 1 ipg_clk is not automatically gated off

31.3.3.10 Interrupt Status Register (IRQSTAT)

An interrupt is generated when at least one of the status bits in this register is set to 1, and the corresponding interrupt enable bit is set in the interrupt signal enable register. All bits are write 1 to clear: writing zeros has no effect. More than one status bit can be cleared with a single register write.

Figure 31-13 shows the register. Table 31-17 describes the register fields.

Address 0xBASE+0x030 (IRQSTAT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DMAE	0	0	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				w1c				w1c		w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W								w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-13. Interrupt Status Register (IRQSTAT)

Table 31-17. Interrupt Status Register Field Descriptions

Field	Description
31–29	Reserved, read as 0
28 DMAE	DMA error. This bit is set to 1, when some error occurs in the internal DMA data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA system address register is the next fetch address where the error occurs. Since any error corrupts the whole data block, the host driver re-starts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0 No DMA error 1 DMA error
27–25	Reserved, read as 0
24 AC12E	Auto CMD12 error. This bit is set to 1 when the eSDHC detects that one of the bits in the auto CMD12 error status register has changed from 0 to 1. This bit is set either when the errors in auto CMD12 occur, or when the auto CMD12 is not executed due to an error in the previous command. 0 No auto CMD12 error 1 Auto CMD12 error
23	Reserved, read as 0
22 DEBE	Data end bit error. Occurs either when the eSDHC detects 0 at the end bit position of read data, which uses the DAT line, or at the end bit position of the CRC. 0 No data end bit error 1 Data end bit error
21 DCE	Data CRC error. Occurs when detecting a CRC error when transferring read data, which uses the DAT line, or when detecting the write CRC status having a value other than 010. 0 No data CRC error 1 Data CRC error

Table 31-17. Interrupt Status Register Field Descriptions (continued)

Field	Description
20 DTOE	Data timeout error. Occurs when detecting one of following time-out conditions. • Busy time-out for R1b,R5b response types • Busy time-out after write CRC status • Read data time-out. 0 No data timeout error 1 Data timeout error
19 CIE	Command index error. Occurs if a command index error occurs in the command response. 0 No command index error 1 Command index error
18 CEBE	Command end bit error. Occurs when detecting that the end bit of a command response is 0. 0 No command end error 1 End bit error generated
17 CCE	Command CRC error. A command CRC error is generated in two cases. • If a CRC error is detected in the command response. In this case, the command timeout error bit in this register remains cleared (indicating no timeout) • If the eSDHC detects a CMD line conflict when the command is issued. This occurs when the eSDHC drives the CMD line to 1, but detects a 0 on the CMD line at the next SD_CLK edge. In this case, the eSDHC aborts the command (stops driving the CMD line), and sets the command timeout error bit in this register to 1. 0 No command CRC error 1 Command CRC error generated.
16 CTOE	Command timeout error. This bit is set if no response is returned within 64 SD_CLK cycles from the end bit of the command. If the eSDHC detects a CMD line conflict this bit is set without waiting for 64 SD_CLK cycles (the command CRC error bit (CCE) bit is also set, as shown in Table 31-20). 0 No command timeout error 1 Command timeout error
15–9	Reserved, read as 0
8 CINT	Card interrupt. This status bit is set when an interrupt signal is detected from the external card. Wakeup is supported: in 1-bit mode, the eSDHC detects the card interrupt without the SD clock. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, which may introducing some delay between the interrupt signal from the SDIO card and the interrupt to the host system. Writing 1 to this bit clears it, but if the interrupt factor from the SDIO card is not cleared then the bit is immediately set again. When CINT has been set to 1, it is the host driver's responsibility to clear the card interrupt signal enable bit in the interrupt signal enable register, so that the eSDHC stops driving the interrupt while it is being serviced. After the card interrupt service is completed, and the interrupt factors in the SDIO card have been reset, then the host driver can write 1 to clear this bit and set the card interrupt signal enable to 1. This causes the eSDHC to restart sampling the interrupt signal. 0 No card interrupt 1 Generate card interrupt
7 CRM	Card removal. This status bit is set if the card inserted bit in the present state register changes from 1 to 0. When writing 1 to this bit to clear it, the host driver is responsible to confirm the value of the card inserted bit in the present state register. This is necessary because it is possible for the card state to change while the host driver clears this bit, and the interrupt event may not be generated. When the CRM bit is cleared, if no card is inserted it is immediately set again: this can be prevented by clearing the card removal status enable bit in interrupt status enable register. 0 Card state unstable or inserted 1 Card removed

Table 31-17. Interrupt Status Register Field Descriptions (continued)

Field	Description
6 CINS	Card insertion. This status bit is set when the card inserted bit in the present state register changes from 0 to 1. When writing 1 to this bit to clear it, the host driver is responsible to confirm the value of the card inserted bit in the present state register. This is necessary because it is possible for the card state to change while the host driver clears this bit, and the interrupt event may not be generated. When the CIN bit is cleared, if a card is inserted it is immediately set again: this can be prevented by clearing the card inserted status enable bit in interrupt status enable register. 0 Card state unstable or removed 1 Card inserted
5 BRR	Buffer read ready. This status bit is set when the buffer read enable bit changes from 0 to 1. Refer to the buffer read enable bit in the present state register description for additional information. 0 Not ready to read buffer 1 Ready to read buffer
4 BWR	Buffer write ready. This status bit is set when the buffer write enable bit changes from 0 to 1. Refer to the buffer write enable bit in the present state register description for additional information. 0 Not ready to write buffer 1 Ready to write buffer:
3 DINT	DMA interrupt. Occurs only when the internal DMA (simple DMA or ADMA) finishes the data transfer successfully. When errors occur during data transfer, this bit is not set: the DMAE bit is set instead. 0 No DMA interrupt 1 DMA interrupt is generated
2 BGE	Block gap event. If the stop at block gap request bit in the protocol control register is set to 1, this bit is set when a read or write transaction is stopped at a block gap. If the stop at block gap request is not set to 1, this bit is not set to 1. For read transactions, this bit is set at the falling edge of the DAT line active status signal (when the transaction is stopped at SD bus timing). Read wait must be supported in order to use this function. For write transactions, this bit is set at the falling edge of the write transfer active status signal (after getting CRC status at SD bus timing). 0 No block gap event 1 Transaction stopped at block gap

Table 31-17. Interrupt Status Register Field Descriptions (continued)

Field	Description
1 TC	Transfer complete. This bit is set when a read or write transfer is completed. For read transactions, this bit is set at the falling edge of the read transfer active status signal. There are two cases in which the transfer complete interrupt is generated: • When a data transfer is completed as specified by the data length (after the last data has been read to the host system). • When data has stopped at the block gap and completed the data transfer by setting the stop at block gap request bit in the Protocol Control register (after valid data has been read to the host system). For write transactions, this bit is set at the falling edge of the DAT line active status signal. There are two cases in which this interrupt is generated: • When the last data is written to the SD card as specified by the data length and the busy signal is released. • When data transfers are stopped at the block gap, by setting the stop at block gap request bit in the protocol control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). 0 Transfer not complete 1 Transfer complete
0 CC	Command complete. This bit is set when the end bit of the command response is received (except auto CMD12). This bit is set to 1 when the command inhibit (CMD) bit in the present state register is cleared. 0 Command not complete 1 Command complete

Table 31-18 shows the eSDHC status associated with different settings of the command timeout error and command complete bits.

Table 31-18. eSDHC Status for Command Timeout Error/Command Complete Bit Combinations

Command Complete Bit	Command Timeout Error Bit	eSDHC Status
0	0	—
0 or 1	1	Response not received within 64 SD_CLK cycles
1	0	Response received

Table 31-19 shows the eSDHC status associated with different settings of the transfer complete and the data timeout error bits.

Table 31-19. eSDHC Status for Data Timeout Error/Transfer Complete Bit Combinations

Transfer Complete Bit	Data Timeout Error Bit	eSDHC Status
0	0	—
0	1	Timeout occurred during transfer
1	0 or 1	Data transfer complete

Table 31-20 shows the eSDHC status associated with different settings of the command CRC error and command timeout error bits.

Table 31-20. eSDHC Status for Command CRC Error/Command Timeout Error Bit Combinations

Command CCR Error Bit	Command Timeout Error Bit	eSDHC Status
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

31.3.3.11 Interrupt Status Enable Register (IRQSTATEN)

Setting bits in this register to 1 enables the corresponding interrupt status register bits to be set by the specified event. If any bit is cleared, the corresponding interrupt status bit is cleared and remains 0.

Figure 31-14 shows the register. Table 31-21 describes the register fields.

Address 0xBASE+0x034 (IRQSTATEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DMAESEN	0	0	0	AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
W																
Reset	0	0	0	1	0	0	0	1	0	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CINTSEN	CRMSEN	CINSSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	1

Figure 31-14. Interrupt Status Enable Register (IRQSTATEN)

Table 31-21. Interrupt Status Enable Register Field Descriptions

Field	Description
31–29	Reserved, read as 0
28 DMAESEN	DMA error status enable 0 Masked 1 Enabled
27–25	Reserved, read as 0
24 AC12ESEN	Auto CMD12 error status enable 0 Masked 1 Enabled
23	Reserved, read as 0
22 DEBESEN	Data end bit error status enable 0 Masked 1 Enabled

Table 31-21. Interrupt Status Enable Register Field Descriptions (continued)

Field	Description
21 DCESEN	Data CRC error status enable 0 Masked 1 Enabled
20 DTOESEN	Data timeout error status enable 0 Masked 1 Enabled
19 CIESEN	Command index error status enable 0 Masked 1 Enabled
18 CEBESEN	Command end bit error status enable 0 Masked 1 Enabled
17 CCESEN	Command CRC error status enable 0 Masked 1 Enabled
16 CTOESEN	Command timeout error status enable 0 Masked 1 Enabled
15–9	Reserved, read as 0
8 CINTSEN	Card interrupt status enable. If this bit is set to 0, the eSDHC clears the interrupt request to the system. the card interrupt detection is stopped when this bit is cleared, and restarted when this bit is set to 1. The host driver is responsible to clear this bit before servicing interrupts, to prevent inadvertent interrupts. After servicing the interrupt, the host driver is responsible to set this bit to 1. 0 Masked 1 Enabled
7 CRMSEN	Card removal status enable 0 Masked 1 Enabled
6 CINSEN	Card insertion status enable 0 Masked 1 Enabled
5 BRRSEN	Buffer read ready status enable 0 Masked 1 Enabled
4 BWRSEN	Buffer write ready status enable 0 Masked 1 Enabled
3 DINTSEN	DMA interrupt status enable 0 Masked 1 Enabled
2 BGESEN	Block gap event status enable 0 Masked 1 Enabled

Table 31-21. Interrupt Status Enable Register Field Descriptions (continued)

Field	Description
1 TCSEN	Transfer complete status enable 0 Masked 1 Enabled
0 CCSEN	Command complete status enable 0 Masked 1 Enabled

NOTE

Depending on the setting of the IABG bit in the protocol control register, eSDHC may be programmed to sample the card interrupt signal during the interrupt period and hold this value in the flip-flop. This can cause a delay between the assertion of the card interrupt signal and notification of the host system.

To detect a CMD line conflict, the host driver sets both the command timeout error status enable bit and the command CRC error status enable bit to 1.

31.3.3.12 Interrupt Signal Enable Register (IRQSIGEN)

This register is used to select the events which are signaled to the host system as interrupts (these status bits all share the same interrupt line). When a bit in this register is set to 1, then setting the corresponding interrupt status register bit to 1 generates an interrupt.

Figure 31-15 shows the register. Table 31-22 describes the register fields.

Address 0xBASE+0x038 (IRQSIGEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DMA EIEN	0	0	0	AC12 EIEN	0	DEBE IEN	DCEI EN	DTOE IEN	CIEIE N	CEBE IEN	CCEIE N	CTO EIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CINTI EN	CRMI EN	CINSI EN	BRRI EN	BWRI EN	DINTI EN	BGEI EN	TCIEN	CCI EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-15. Interrupt Signal Enable Register (IRQSIGEN)

Table 31-22. Interrupt Signal Enable Register Field Descriptions

Field	Description
31–29	Reserved, read as 0
28 DMAEIEN	DMA error interrupt enable 0 Masked 1 Enable
27–25	Reserved, read as 0
24 AC12EIEN	Auto CMD12 error interrupt enable 0 Masked 1 Enabled
23	Reserved, read as 0
22 DEBEIEN	Data end bit error interrupt enable 0 Masked 1 Enabled
21 DCEIEN	Data CRC error interrupt enable 0 Masked 1 Enabled
20 DTOEIEN	Data timeout error interrupt enable 0 Masked 1 Enabled
19 CIEIEN	Command index error interrupt enable 0 Masked 1 Enabled
18 CEBEIEN	Command end bit error interrupt enable 0 Masked 1 Enabled
17 CCEIEN	Command CRC error interrupt enable 0 Masked 1 Enabled
16 CTOEIEN	Command timeout error interrupt enable 0 Masked 1 Enabled
15–9	Reserved, read as 0
8 CINTIEN	Card interrupt interrupt enable 0 Masked 1 Enabled
7 CRMIEN	Card removal interrupt enable 0 Masked 1 Enabled
6 CINIEN	Card insertion interrupt enable 0 Masked 1 Enabled
5 BRRIEN	Buffer read ready interrupt enable 0 Masked 1 Enabled

Table 31-22. Interrupt Signal Enable Register Field Descriptions (continued)

Field	Description
4 BWRIEN	Buffer write ready interrupt enable 0 Masked 1 Enabled
3 DINTIEN	DMA interrupt enable 0 Masked 1 Enabled
2 BGEIEN	Block gap event interrupt enable 0 Masked 1 Enabled
1 TCIEN	Transfer complete interrupt enable 0 Masked 1 Enabled
0 CCIEN	Command complete interrupt enable 0 Masked 1 Enabled

31.3.3.13 Auto CMD12 Error Status Register (AUTOC12ERR)

When the auto CMD12 error status bit in the status register is set to 1, the host driver checks this register to identify the source of auto CMD12 errors. This register is valid only when the auto CMD12 error status bit in the status register is set to 1.

Figure 31-16 shows the register. Table 31-23 describes the register fields.

Address 0xBASE+0x03C (AUTOC12ERR) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	CNIBAC12E	0	0	AC12IE	AC12CE	AC12EBE	AC12TOE	AC12NE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-16. Auto CMD12 Error Status Register (AUTOC12ERR)

Table 31-23. Auto CMD12 Error Status Register Field Descriptions

Field	Description
31–8	Reserved, read as 0
7 CNIBAC12E	Command not issued by auto CMD12 error. This bit is set to 1 when CMD_wo_DAT is not executed due to an auto CMD12 error (D04–D01) in this register. 0 No error 1 Not issued
6–5	Reserved, read as 0
4 AC12IE	Auto CMD12 index error. Indicates that a command index error occurred in response to a command. 0 No error 1 Error, the CMD index in response is not CMD12
3 AC12CE	Auto CMD12 CRC error. Indicates a CRC error in the command response. 0 No CRC error 1 CRC Error Met in auto CMD12 Response
2 AC12EBE	Auto CMD12 end bit error. indicates the end bit of command response is 0 which should be 1. 0 No error 1 End Bit Error Generated
1 AC12TOE	Auto CMD12 timeout error. Indicates that no response is returned within 64 SD_CLK cycles after the end bit of the command. If this bit is set to1, then the other error status bits (2–4) are not valid. 0 No error 1 Time out
0 AC12NE	Auto CMD12 not executed. This bit is set to 1 when the eSDHC cannot issue the auto CMD12 to stop a memory multi-block data transfer due to some error. In case the memory multi-block data transfer is not started due to a command error, this bit is not set because it is not necessary to issue an auto CMD12. If this bit is set to 1, other error status bits (1-4) have no meaning. 0 Executed 1 Not executed

Table 31-24 shows error types associated with different auto CMD12 CRC error and auto CMD12 command timeout error bit settings.

Table 31-24. Command CRC Error and Command Timeout Error Bit Settings and Error Types

Auto CMD12 CRC Error Bit	Auto CMD12 Timeout Error Bit	Type of Error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

When issuing auto CMD12 commands, the eSDHC sets bits in the auto CMD12 error status register according to the following steps:

1. Before issuing an auto CMD12 command, the eSDHC sets the auto CMD12 not executed bit (bit 0) of the auto CMD12 error status register if the auto CMD12 cannot be issued due to an error in the previous command.
2. If the auto CMD12 is issued, then the auto CMD12 not executed bit (bit 0) is cleared.

3. At the end bit of an auto CMD12 response, the eSDHC checks errors corresponding to bits 0–4 (AC12IE, AC12CE, AC12EBE, and AC12TOE) and sets the bit if the corresponding error is detected.
4. Before reading the command not issued by auto CMD12 error bit (bit 7) the eSDHC sets bit 7 to 1 if there is a command that cannot be issued, and clears bit 7 if there is no command to issue.

Auto CMD12 errors and writes to the command register are asynchronous. After an auto CMD12 command, bit 7 is sampled when the driver is not writing to the command register. The driver can avoid problems by setting the AC12E bit in the interrupt status register before reading this register. An auto CMD12 error interrupt is generated when one of the error bits (0-4) is set to 1. The command not issued by auto CMD12 error does not generate an interrupt.

31.3.3.14 Host Controller Capabilities Register (HOSTCAPBLT)

This register provides the host driver with information specific to the eSDHC implementation. The value in this register is the same as at power-on reset, and does not change during a software reset. Any write to this register is ignored.

Figure 31-17 shows the register. Table 31-25 describes the register fields.

Address 0xBASE+0x040 (HOSTCAPBL1T) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL[2:0]		
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-17. Host Controller Capabilities Register (HOSTCAPBLT)

Table 31-25. Host Capabilities Register Field Descriptions

Field	Description
31–27	Reserved, read as 0
26 VS18	Voltage support 1.8 V. This bit's setting depends on the host system ability. 0 1.8V not supported 1 1.8V supported
25 VS30	Voltage support 3.0 V. This bit's setting depends on the host system ability. 0 3.0 V not supported 1 3.0 V supported

Table 31-25. Host Capabilities Register Field Descriptions (continued)

Field	Description
24 VS33	Voltage support 3.3 V. This bit's setting depends on the host system ability. 0 3.3 V not supported 1 3.3 V supported
23 SRS	Suspend/Resume support. This bit indicates whether the eSDHC supports Suspend/Resume functionality. If this bit is 0, the Suspend and Resume mechanism, as well as the Read Wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0 Not supported 1 Supported
22 DMAS	DMA support. This bit indicates whether the eSDHC is capable of using the internal DMA to transfer data between system memory and the data buffer directly. 0 DMA not supported 1 DMA supported
21 HSS	High-speed mode support. This bit indicates whether the eSDHC supports High-speed mode and the Host System can supply a SD clock frequency from 25 MHz to 50 MHz. 0 High-speed not supported 1 High-speed supported
20 ADMAS	ADMA support. This bit indicates whether the eSDHC supports the ADMA feature. 0 Advanced DMA not supported 1 Advanced DMA supported
19	Reserved, read as 0
18–16 MBL[2:0]	Maximum block length. This value indicates the maximum block size that the host driver can read and write to the eSDHC's buffer. The buffer transfers blocks up to this size without wait cycles. 000 512 bytes 001 1024 bytes 010 2048 bytes 011 4096 bytes
15–0	Reserved, read as 0

31.3.3.15 Watermark Level Register (WML)

This register configures watermark levels (FIFO thresholds) and burst lengths for write and read. Watermark levels can range from 1–128 words; burst lengths can range from 1–31 words.

Figure 31-18 shows the register. Table 31-26 describes the register fields.

Address 0xBASE+0x044 (WML) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	WR_BRST_LEN[3:0]					WR_WML[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	RD_BRST_LEN[3:0]					RD_WML[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Figure 31-18. Watermark Level Register (WML)

Table 31-26. Watermark Level Register Field Descriptions

Field	Description
31–29	Reserved, read as 0
28–24 WR_BRST_LEN[4:0]	Write burst length. The number of words the eSDHC writes in a single burst. The write burst length must not exceed the write watermark level, and all bursts within a watermark level transfer are in back-to-back mode. This field resets to 0b01000. The field cannot be cleared: writing 0 to this field results in the reset value 0b01000. **Note:** Due to system restrictions, the actual burst length does not exceed 16.
23–16 WR_WML[7:0]	Write watermark level. The number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also the number of words in a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128. The reset write watermark level is 16.
15–13	Reserved, read as 0
12–8 RD_BRST_LEN[4:0]	Read burst length. The number of words the eSDHC reads in a single burst. The read burst length must not exceed the read watermark level, and all bursts within a watermark level transfer are in back-to-back mode. This field resets to 0b01000. The field cannot be cleared: writing 0 to this field results in the reset value 0b01000. **Note:** Due to system restrictions, the actual burst length does not exceed 16.
7–0 RD_WML[7:0]	Read watermark level. The number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also the number of words in a sequence of read bursts in back-to-back mode. The maximum legal value for the read watermark level is 128. The reset read watermark level is 16.

31.3.3.16 Force Event Register (FEVT)

The force event register is not a physically-implemented register, but rather an address where the interrupt status register can be written if the corresponding bit of the interrupt status enable register is set. Writing 1’s to this register sets the corresponding bits in the interrupt status register. This register is a write-only register: reads always return zero. Writing zero to the register has no effect.

When writing to this register to change status bits in the interrupt status register, the driver is responsible to ensure that the ipg_clk is active by setting the IPGEN bit in the system control register.

Figure 31-19 shows the register. Table 31-27 describes the register fields.

Address 0xBASE+0x050 (FEVT) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	FEVTCINT			FEVTDMAE				FEVTAC12E		FEVTDEBE	FEVTDCE	FEVTDTOE	FEVTCIE	FEVTCEBE	FEVTCCE	FEVTTCTOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W									FEVTCNIBAC12E			FEVTAC12IE	FEVTAC12EBE	FEVTAC12CE	FEVTAC12TOE	FEVTAC12NE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-19. Force Event Register (FEVT)

Table 31-27. Force Event Register Field Descriptions

Field	Description
31 FEVTCINT	Force event card interrupt. Writing 1 to this bit generates a short low-level pulse on the internal DAT[1] line, as if a self clearing interrupt was received from the external card. If enabled, the CINT bit is set and the interrupt service routine responds to this interrupt as a normal interrupt from the external card. It is not necessary for he interrupt service routine to poll the card interrupt factor, since the interrupt is self cleared.
30–29	Reserved, read as 0
28 FEVTDMAE	Force Event DMA Error. Forces the DMAE bit of interrupt status register to be set
27–25	Reserved, read as 0
24 FEVTAC12E	Force event auto CMD12 error. Forces the AC12E bit of interrupt status register to be set
23	Reserved, read as 0
22 FEVTDEBE	Force event data end bit error. Forces the DEBE bit of interrupt status register to be set
21 FEVTDCE	Force event data CRC error. Forces the DCE bit of interrupt status register to be set
20 FEVTDTOE	Force event data time out error. Force the DTOE bit of interrupt status register to be set

Table 31-27. Force Event Register Field Descriptions (continued)

Field	Description
19 FEVTCIE	Force event command index error. Forces the CCE bit of interrupt status register to be set
18 FEVTCEBE	Force event command end bit error. Forces the CEBE bit of interrupt status register to be set
17 FEVTCCE	Force event command CRC error. Forces the CCE bit of interrupt status register to be set
16 FEVTCCE	Force event command time out error. Forces the CTOE bit of interrupt status register to be set
15–8	Reserved, read as 0
7 FEVTCNIBAC12E	Force event command not executed by auto CMD12 error. Forces the CNIBAC12E bit in the Auto CMD12 Error Status Register to be set.
6–5	Reserved, read as 0
4 FEVTAC12IE	Force event auto CMD12 Index error. Forces the AC12IE bit in the auto CMD12 error status register to be set.
3 FEVTAC12EBE	Force event auto CMD12 end bit error. Forces the AC12EBE bit in the auto CMD12 error status register to be set.
2 FEVTAC12CE	Force event auto CMD12 CRC error. Forces the AC12CE bit in the auto CMD12 error status register to be set.
1 FEVTAC12TOE	Force event auto CMD12 time out error. Forces the AC12TOE bit in the auto CMD12 error status register to be set.
0 FEVTAC12NE	Force event auto CMD12 not executed. Forces the AC12NE bit in the auto CMD12 error status register to be set.

31.3.3.17 ADMA Error Status Register (ADMAES

When an ADMA error interrupt occurs, the ADMA error status (ADMAES) field in this register indicates the ADMA error state. To recover from the error, the ADMAES field together with the system address register can be used to determine the address of the error descriptor (see the ADMAES field description in Table 31-28).

In case of a write operation, the host driver should use the ACMD22 to get the number of the written blocks, rather than using this information, since unwritten data may exist in the host controller.

The host controller generates the ADMA error interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the valid bit of the error descriptor.

Figure 31-20 shows the register. Table 31-28 describes the register fields

Address 0xBASE+0x054 (ADMAES) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	ADMADCE	ADMALME	ADMAES[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 31-20. ADMA Error Status Register (ADMAES)

Table 31-28. ADMA Error Status Register Field Descriptions

Field	Description
31–4	Reserved, read as 0
3 ADMADCE	ADMA Descriptor Error. This error occurs when an invalid descriptor is fetched by ADMA: 0 No Error 1 Error
2 ADMALME	ADMA length mismatch error. This error occurs in the following two cases: • The block count enable bit (BCEN) in the transfer type register is set, and the total data length specified by the descriptor table is different from that specified by the block count and block size. • The BCEN bit is cleared, and the total data length is not a multiple of the block size 0 No error 1 Error
1–0 ADMAES	ADMA error state. This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. ADMAES settings and corresponding error states and ADMA system address register contents are shown in the following table.

ADMAE Setting	ADMA Error State	ADMA System Address Register Contents
00	ST_STOP (stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (change address)	No ADMA error is generated
11	ST_TFR (transfer data)	Holds the address of the next executable descriptor command

31.3.3.18 ADMA System Address Register (ADSADDR)

This register contains the physical system memory address used for ADMA transfers. Figure 31-21 shows the register. Table 31-29 describes the register fields.

Address 0xBASE+0x058 (ADSADDR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADS_ADDR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADS_ADDR[15:0]														0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 31-21. ADMA System Address Register (ADSADDR)

Table 31-29. ADMA System Address Register Field Descriptions

Field	Description
31–2 ADS_ADDR[31:0]	ADMA system address. This register holds the address of the executing command word in the descriptor table. At the start of ADMA transfer, the host driver sets the start address of the descriptor table. The ADMA engine increments this register address every fetch of a descriptor command. When the ADMA is stopped at the block gap, this register indicates the address of the next executable descriptor command. When the ADMA error interrupt is generated, this register holds the valid descriptor address or the next executable descriptor command, depending on the ADMA state (see Table 31-28). The lower 2 bits of this register are tied to '0' so the ADMA address is always word aligned. This register supports dynamic address reflection: when the transfer complete (TC) bit is set in the interrupt status register it automatically alters the value of internal address counter, so software cannot change this register when TC bit is set. Additional software restrictions are listed in Section 31.7, Software Restrictions."
1–0	Reserved, read as 0

31.3.3.19 Vendor Specific Register (VENDOR)

This register contains the vendor-specific control/status settings.

Figure 31-22 shows the register. Table 31-30 describes the register fields.

Address 0xBASE+0x0C0 (VENDOR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	DBG_SEL[3:0]				INT_ST_VAL[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EXT_DMA_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 31-22. Vendor Specific Register (VENDOR)

Table 31-30. Vendor Specific Register Field Descriptions

Field	Description
31–28	Reserved, read as 0
27–24 DBG_SEL[3:0]	Debug select. Select the internal submodule to show its internal state value.
23–16 INT_ST_VAL[7:0]	Internal state value. Internal state value, reflecting the corresponding state value selected by Debug Select field. This field is read-only and write to this field does not have effect.
15–1	Reserved, read as 0
0 EXT_DMA_EN	External DMA request enable. Enables requests to external DMA. When the internal DMA (either simple DMA or advanced DMA) is not in use and this bit is set to 1, then the eSDHC sends out a DMA request when the internal buffer is ready. Clearing this bit disables the external DMA request: this can be useful in CPU polling mode. By default, this bit is set. 0 In any scenario, eSDHC does not send out external DMA request. 1 When internal DMA is not active, an external DMA request is sent out (reset value)

31.3.3.20 Host Controller Version (HOSTVER)

This register contains the vendor host controller version information. All bits are read-only with the same value as on power-on reset.

Figure 31-23 shows the register. Table 31-31 describes the register fields.

Offset 0xBASE+0x0FC (HOSTVER) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VVN[7:0]								SVN[7:0]							
W																
Reset	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1

Figure 31-23. Host Controller Version Register (HOSTVER)

Table 31-31. Host Controller Version Register Field Descriptions

Field	Description
31–16	Reserved, read as 0
15–8 VVN[7:0]	Vendor version number. These status bits are reserved for the vendor version number. The host driver does not use this status. 0x00 Freescale eSDHC Version 1.0 0x10 Freescale eSDHC Version 2.0 0x11 Freescale eSDHC Version 2.1 0x12 Freescale eSDHC Version 2.2 others) Reserved
7–0 SVN[7:0]	Specification version number. These status bits indicate the Host Controller Specification version. 0x01 SD Host Specification Version 2.0, supports test event register and ADMA. All others) Reserved

31.4 Functional Description

The following subsections provide a brief functional description of the major system blocks, including the data buffer, DMA AHB interface, register bank and IP bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

31.4.1 Data Buffer

The eSDHC uses a configurable data buffer to optimize data transfer between the system bus (IP Bus or AHB bus) in the master clock (hclk) domain, and the SD card in the IP peripheral source clock (ipg_perclk) domain

Figure 31-24 shows the buffer scheme. The buffer is used as temporary storage for data being transferred between the host system and the card. The watermark levels for read and write are both configurable, from 1–128 words inclusive. The burst lengths for read and write are also configurable, from 1–16 words

inclusive (the burst length field of the watermark level register can range from 1–31, but actual burst lengths greater than 16 are not supported).

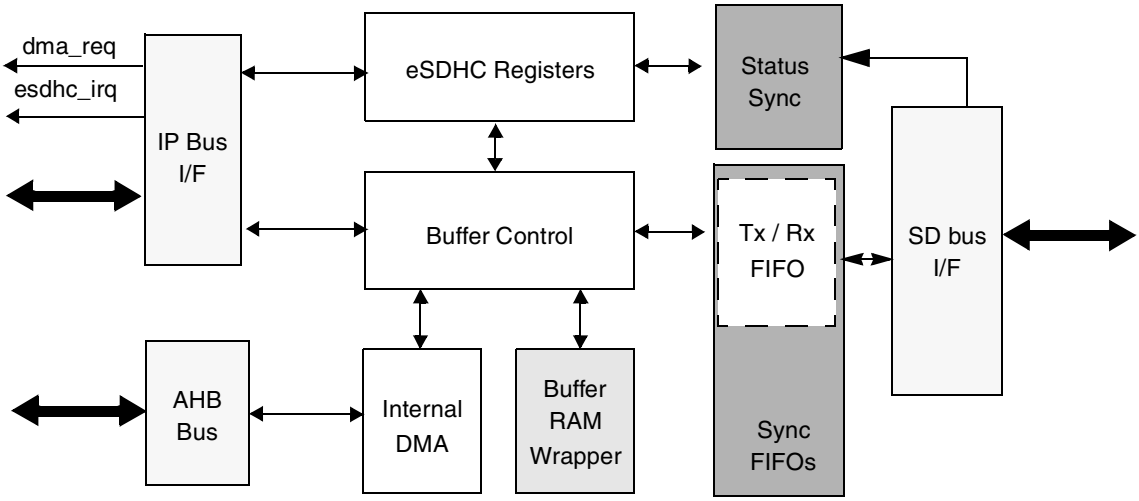


Figure 31-24. eSDHC Buffer Scheme

31.4.1.1 Data Buffer Transfer Modes

There are three transfer modes to access the data buffer:

- External DMA mode
- CPU polling mode
- Internal DMA mode (includes simple and advanced DMA accesses)

These transfer modes are explained in the following subsections

31.4.1.1.1 External DMA Mode

External DMA uses the IP bus. Using external DMA requires that DMAEN bit in the transfer type register be cleared when the DMA request is sent.

For read operations, the eSDHC sends an external DMA request (by asserting the signal esdhc_dreq_b to 0) when the number of words received in the buffer meets or exceeds the read watermark value (RD_WML in the watermark level register). The request is immediately negated when there is an access on the buffer data port register. If the number of words in the buffer after the current burst still meets or exceeds RD_WML value, the DMA request is reasserted, with one idle cycle between successive requests.

Write operations in external DMA mode are similar to read operations, except the write watermark value WR_WML is used instead of RD_WML to determine whether there are sufficiently many empty words in the buffer before the eSDHC sends the external DMA request.

31.4.1.1.2 CPU Polling Mode

CPU polling mode is similar to external DMA mode in that it uses the IP bus, and requires the DMAEN bit in the transfer type register be cleared.

CPU polling mode requires that the BRRIEN bit be set to 1 in the interrupt signal enable register. It differs from external DMA mode in that rather than relying on an external DMA request the host driver polls the BRR bit in the interrupt status register, as follows:

For read operations, when the number of words received in the buffer meets or exceeds the read watermark value (RD_WML field in the watermark level register) the eSDHC sets the BRR bit in the interrupt status register to 1 (this occurs simultaneously with sending the external DMA request). The host driver is responsible to poll the BRR bit, and when it detects that BRR is set it reads the buffer data port register to fetch RD_WML words from the buffer.

Write operation in CPU polling mode requires setting the BWRIEN to 1 in the interrupt signal enable register. Write operations are similar to read operations, except that the write watermark level WR_WML is used instead of RD_WML to determine whether there are sufficiently many empty words in the buffer, and the BWR bit is polled instead of BRR.

31.4.1.1.3 Internal DMA Mode

Internal DMA accesses utilize the AHB bus, and requires that the DMAEN bit in the transfer type register be set to 1 when the DMA request is sent. Unlike external DMA or CPU polling mode, the external DMA request is never sent out.

Internal DMA Read Operations

For internal DMA read operations, when the number of words in the buffer meets or exceeds the watermark level (RD_WML field in the watermark level register) the internal DMA starts fetching data over the AHB bus. The burst type is always INCR mode, and the burst length depends on the shortest of following factors:

- Burst length, configured in the burst length field of the watermark level register
- Watermark level boundary
- Remaining number of words in the current block
- Data boundary, configured in the current descriptor (if the ADMA is active)
- 1 Kbyte address boundary defined in the AHB protocol

When internal DMA is used, if no error is encountered the eSDHC does not inform the system before the required number of bytes are transferred. When an error occurs during the data transfer, the eSDHC aborts the data transfer and abandon the current block. The host driver is responsible to read the contents of the DMA system address register to find the starting address of the abandoned data block. If the current data transfer is in multi block mode, the eSDHC does not automatically send CMD12, even though the AC12EN bit in the transfer type register is set. Instead, it is the host drivers responsibility to send CMD12 and restart the read operation from that address. It is recommended that a software reset for data be applied before the transfer is restarted after error recovery.

The eSDHC does not start data transmission until the number of words in the buffer meets or exceeds the read watermark level RD_WML. If the buffer is full and the host system does not read data in time, the eSDHC stops SD_CLK to avoid a data buffer overrun.

Internal DMA Write Operations

Write operations in internal DMA mode are similar to read operations, except that the write watermark level WR_WML is used to determine whether there are sufficiently many empty words in the buffer.The eSDHC does not start data transmission until the number of words set in the WR_WML register can be held in the buffer. If the buffer is empty and the host system does not write data in time, the eSDHC stops SD_CLK to avoid a data buffer under-run situation.

31.4.1.2 Data Addressing and Byte Ordering

Sequential and contiguous access is necessary to update the pointer address value correctly. Random or skipped access is not possible. On reset, byte ordering is set to little endian mode. The actually byte order is swapped inside the buffer according to the endian mode configured by software, as shown in Figure 31-25 and Figure 31-26. For a host write operation, byte order is swapped after data is fetched from the buffer and before it is sent to the SD bus. For a host read operation, byte order is swapped before the data is stored into the buffer.

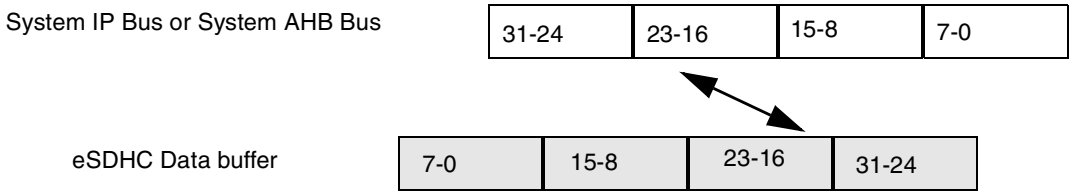


Figure 31-25. Data Swap between System Bus and eSDHC Data Buffer in Byte Little Endian Mode

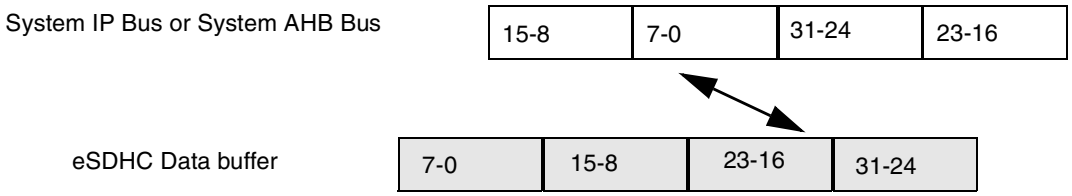


Figure 31-26. Data Swap Between System Bus and eSDHC Data Buffer in Half Word Big Endian Mode

31.4.1.3 Data Transfer Parameter Settings

In the eSDHC, the data buffer holds up to 128 4-byte words. The watermark levels for write and read can be configured from 1–128 words inclusive. For both read and write, the burst length, can be from 1–16 words inclusive (the burst length field in the watermark level ranges from 1–31, but actual burst lengths greater than 16 are not supported). The host driver is responsible to configure these values according to the system situation and requirements.

During a multi-block data transfer, the block size may be set to any value between 1 and 4096 bytes. However, additional restrictions may be imposed by the external card, which may not support large block sizes and/or partial block accesses.

31.4.1.3.1 Dividing Large Data Transfers

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multi-block transfer must be an integer multiple of the block size. If the total data length is not a multiple of the block size then there are two ways to transfer the data, depending on the function and the card design:

- The host driver splits the transaction, and fractional block portion of data is transferred using a second (single block) transfer.
- Dummy data is added to fill the last block. In this case, the card must manage the removal of the dummy data.

Figure 31-27 shows an example of a large data transfer using a WLAN SDIO card that only supports block sizes up to 64 bytes. In this case, 544 bytes are divided into eight 64-byte blocks plus 32 bytes, which are transferred using two separate CMD53 commands.

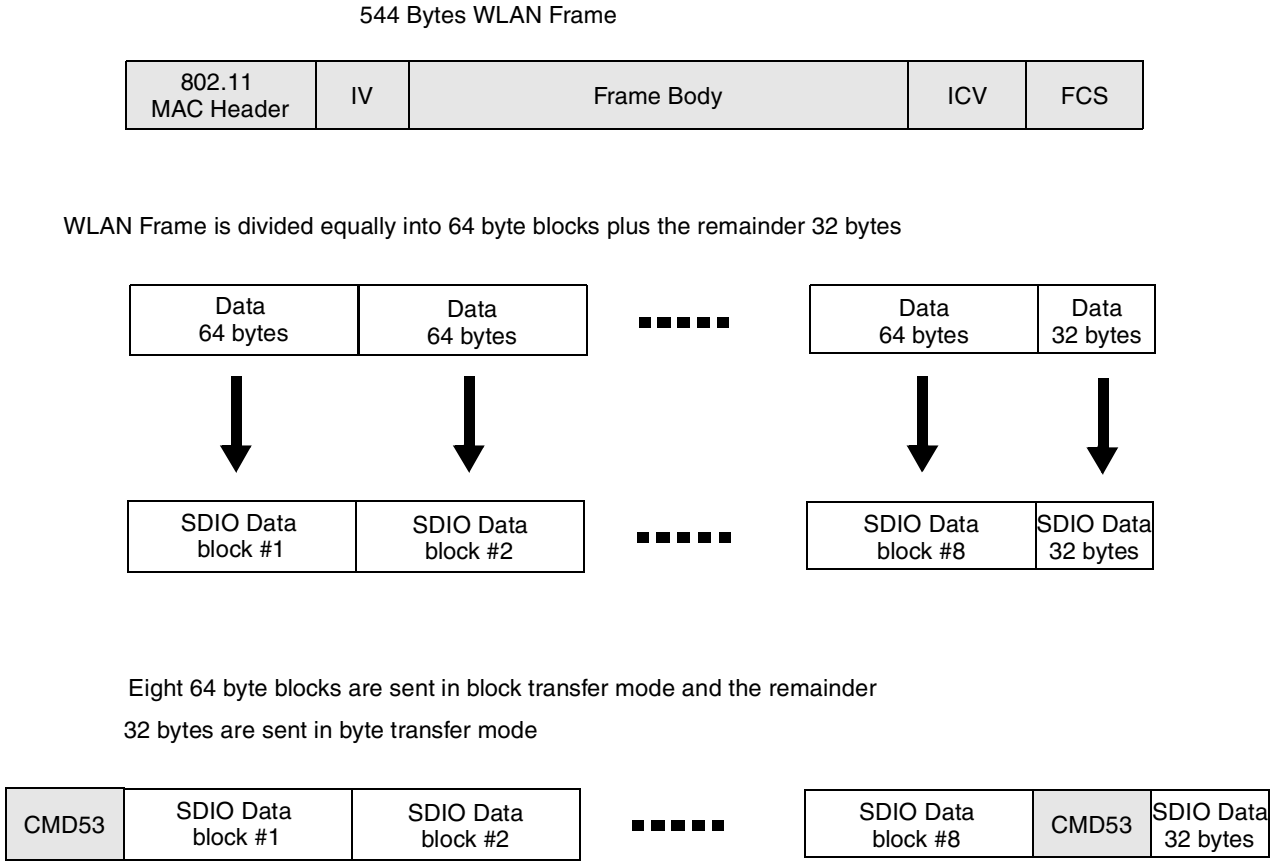


Figure 31-27. Example for Dividing Large Data Transfers

31.4.1.4 Data Transfer Parameters for External DMA Requests

[In external DMA transfer mode, since the DMA burst length cannot change during a data transfer it is necessary that the watermark level (read or write) must be a divisor of the block size (in word units): otherwise, transferring of the block may cause buffer underrun (for reads) or overrun (for writes). This implies for example that a block size of 512 bytes (128 words) requires that read and write watermark levels must be a power of two between 1 and 128 inclusive.

In CPU polling mode there are no such restrictions on the watermark levels, because the last access in the block transfer can be controlled by software. The watermark levels can even be larger than the block size (but no greater than 128 words). The actual number of bytes in each transfer is controlled by software, and does not exceed the block size.

Non-word–aligned block sizes are supported in CPU polling mode, as long as the card supports that block size. For example, the block size can be set at 31 bytes (if supported by the card), and the watermark level and burst length can take any allowed value. This is because the software transfers 8 words, and the eSDHC also sets the BRR or BWR bits (for reads or writes, respectively) when the remaining data satisfies watermark level conditions. Even though 8 words are transferred via the data port register, the eSDHC transfers only 31 bytes over the SD bus, as required by the BLKSIZE bits. In data transfers with non-word aligned block sizes, the endian mode should be set carefully, or invalid data can be transferred to/from the card.

31.4.1.5 Data Transfer Parameters for Internal DMA Requests

Internal DMA data transfers are in block units, and the host driver is responsible to set the watermark level as the minimum remaining number of words remaining in a block following a series of burst transfers. For instance, consider a multi-block read with block size 31 bytes and burst length set to 6 (4-byte) words. After the first burst transfer, there are 7 (= 31 – 4 × 6) bytes remaining to be read from the first block: the remaining data occupies 2 words. The host driver sets the read watermark level as 2 words, so that another DMA request is sent if there are 2 or more words in the buffer (which might contain some data from the next block). The eSDHC reads 2 words out of the buffer, which include 7 valid bytes and 1 stuff byte.

The DMA burst length for the internal DMA engine can be from 1 to 16 words inclusive, just as in CPU polling mode. The actual burst length for the DMA depends on the smaller of the configured burst length and the remaining words of the current block. In the above example with block size 31 bytes and burst length set to 6, the actual burst lengths alternate between 6 and 2 words. The host driver is responsible to take this variable burst length into account. One option is to configure the burst length as a divisor of the block size, so that the burst length does not need to vary.

31.4.2 DMA AHB Interface

Figure 31-28 shows the DMA AHB interface block. The internal DMA AHB interface includes a DMA engine and the AHB master.

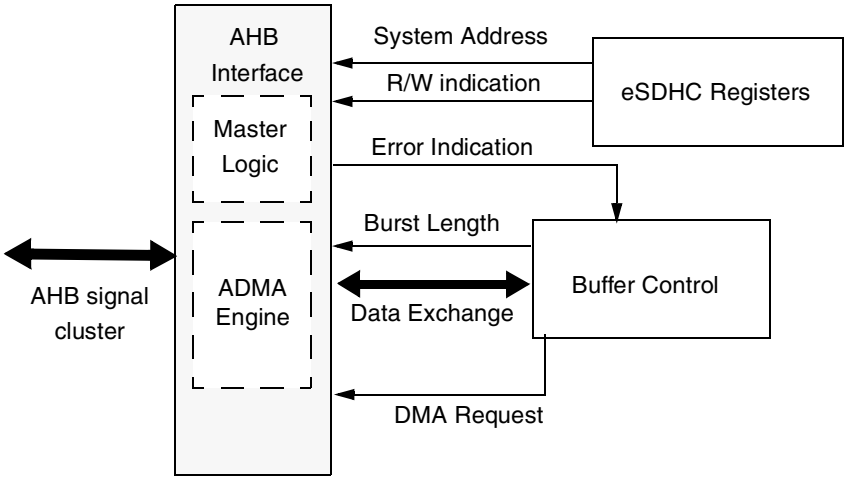


Figure 31-28. DMA AHB Interface Block

31.4.2.1 Internal DMA Requests

Internal DMA transfers are enabled by setting the DMAEN bit in the transfer type register. The external DMA request signal (esdhc_dreq_b) is disabled: however, the BRR and BWR bits in the interrupt status register are still valid, as long as the BRRSEN and BWRSEN bits have been set in the interrupt status enable register. See Section 31.4.1.1.3, Internal DMA Mode," for more details about internal DMA transfers.

If the watermark level requirement is met, the data buffer block sends a DMA request to the AHB interface. There is a delay in the internal DMA engine's response that depends on the system AHB bus loading and the priority assigned to the eSDHC. The DMA engine does not respond to a request during burst transfers, but is ready to serve as soon as the burst is over. The data buffer negates the request after the buffer is accessed. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and if the watermark level requirement is satisfied another DMA request is sent.

31.4.2.2 AHB Master Interface

If the internal AHB DMA engine fails during the data transfer, the following actions occur:

1. The DMA engine stops the transfer and goes to the idle state
2. The internal data buffer stops accepting incoming data.
3. The DMAE bit in the interrupt status register is set to inform the driver.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and reads the DS_ADDR bits of the DMA system address register to get the starting address of the corrupted block. After the DMA error is fixed, the software applies a data reset and restarts the transfer from this address to recover the corrupted block.

31.4.2.3 ADMA Engine

The ADMA (Advanced DMA) transfer algorithm is defined in the SD Host Controller Standard. In contrast with simple DMA transfers, which generate a DMA interrupt at each page boundary so that the host driver can program a new system address, ADMA transfers define a programmable descriptor table in the system memory. The host driver can then calculate the system address at the page boundary and program the descriptor table before executing ADMA. This enables higher speed DMA transfers, because host intervention is not needed during long DMA-based data transfers.

The eSDHC implements two types of ADMA: ADMA1 and ADMA2. ADMA1 supports data transfers of 4-KB–aligned data in system memory. ADMA2 supports data transfers of any size from any location in system memory. ADMA1 and ADMA2 have different descriptor table formats, which are described in Section 31.4.2.3.1, ADMA1 Descriptor Tables".

The ADMA engine recognizes all descriptor types defined in the SD Host Controller Standard. The ADMA1 and ADMA2 descriptors are described in Section 31.4.2.3.1, ADMA1 Descriptor Tables" and Section 31.4.2.3.2, ADMA2 Descriptor Tables," respectively.

31.4.2.3.1 ADMA1 Descriptor Tables

ADMA1 includes the following descriptor types:

- No operation (Nop): No operation is performed, pointer passes to the next descriptor
- Set data length (Set): Specifies data length for the next data transfer
- Transfer data (Tran): Initiates data transfer
- Link descriptor (Link): Links to the next descriptor in the table, at a non-consecutive location in system memory

Descriptors also contain interrupt, end, and valid/invalid flag bits which inform the ADMA engine of the descriptor's status.

Figure 31-29 shows the ADMA1 descriptor format. Table 31-32 describes the ADMA1 field settings corresponding to each descriptor type, and Table 31-33 describes the interrupt, end, and valid/invalid flag bits.

Address/Page Field		Address/Page Field		Attribute Fields					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act2	Act1	0	Int	End	Valid

Figure 31-29. ADMA1 Descriptor Format

Table 31-32. ADMA1 Descriptor Type Field Settings

Descriptor Type Abbreviation	Descriptor Type Name	Descriptor Field Settings			
		31:12 Address/Page	27:12 Address/Page	5 Act2	4 Act1
Nop	No Operation	Don't Care		0	0
Set	Set Data Length	0000	Data Length	0	1
Tran	Transfer Data	Data Address		1	0
Link	Link Descriptor	Descriptor Address		1	1

Table 31-33. ADMA1 Descriptor Flag Bit Descriptions

Flag Bit	Description
Int (Bit 2)	Int =1 generates a DMA interrupt when this descriptor is processed.
End (Bit 1)	End = 1 indicates current descriptor is the last one in the table.
Valid (Bit 0)	Valid =1 indicates the descriptor is effective. If Valid = 0, the ADMA engine generates an ADMA error interrupt and stops the ADMA.

Figure 31-30 shows the ADMA1 descriptor table structure and its accesses by the ADMA engine. Every Tran type descriptor triggers a data transfer, with data length specified by the most recent Set type descriptor. If there is no Set type descriptor after the previous Trans descriptor, the data length from the previous transfer is used. If no Set descriptor has been read, the data length is 0.

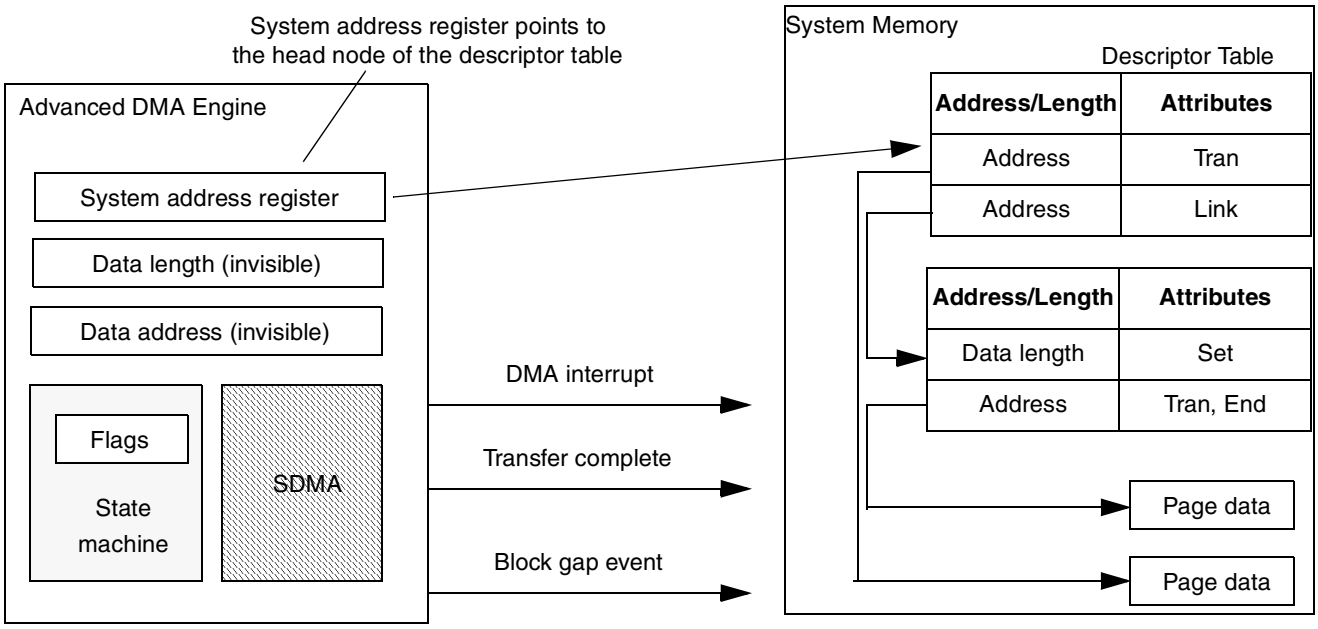


Figure 31-30. ADMA1 Descriptor Table and Access by the ADMA Engine

31.4.2.3.2 ADMA2 Descriptor Tables

ADMA2 includes the following descriptor types:

- No operation (Nop): No operation is performed, pointer passes to the next descriptor
- Reserved (Rsv): No operation is performed, passes to the next descriptor (same as Nop)
- Transfer data (Tran): Transfers data with address and length set in this descriptor line
- Link descriptor (Link): Links to the next descriptor in the table, at a non-consecutive location in system memory

Descriptors also contain interrupt, end and valid/invalid flag bits which inform the ADMA engine of the descriptor's status.

Figure 31-31 shows the ADMA2 descriptor format. Table 31-32 describes the ADMA2 Act*n* bit settings corresponding to each descriptor type, and Table 31-33 describes the interrupt, end, and valid/invalid flag bits.

Address Field		Length Field		Reserved		Attribute Fields					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit address		16-bit length		0000000000		Act2	Act1	0	Int	End	Valid

Figure 31-31. ADMA2 Descriptor Format

Table 31-34. ADMA2 Descriptor Type Field Settings

Descriptor Type Abbreviation	Descriptor Type Name	Act*n* Bit Settings		Descriptor Operation
		5 Act2	4 Act1	
Nop	No Operation	0	0	No operation: read this line and go on to the next
Rsv	Reserved	0	1	No operation: read this line and go on to the next
Tran	Transfer data	1	0	Transfer data with address and length set in this descriptor line
Link	Link descriptor	1	1	Links to the next descriptor in the table, at a non-consecutive location in system memory

Table 31-35. ADMA2 Descriptor Flag Bit Descriptions

Flag Bit	Description
Int (Bit 2)	Int =1 generates a DMA interrupt when this descriptor is processed.
End (Bit 1)	End = 1 indicates current descriptor is the last one in the table.
Valid (Bit 0)	Valid =1 indicates the descriptor is effective. If Valid = 0, the ADMA engine generates an ADMA error interrupt and stops the ADMA.

Figure 31-32 shows the ADMA2 descriptor table structure and its accesses by the ADMA engine. The ADMA engine first processes the descriptor's lower 32 bits, before reading the upper 32 bits. If the Valid flag (bit 0) of the descriptor is 0, the upper 32 bits are ignored. The address field contains word-aligned addresses, so the lowest 2 bits are always set to 0. The data length is specified in bytes.

ADMA2 read/write operations are initiated by Tran descriptors, and use the data length and address specified in the descriptor itself.

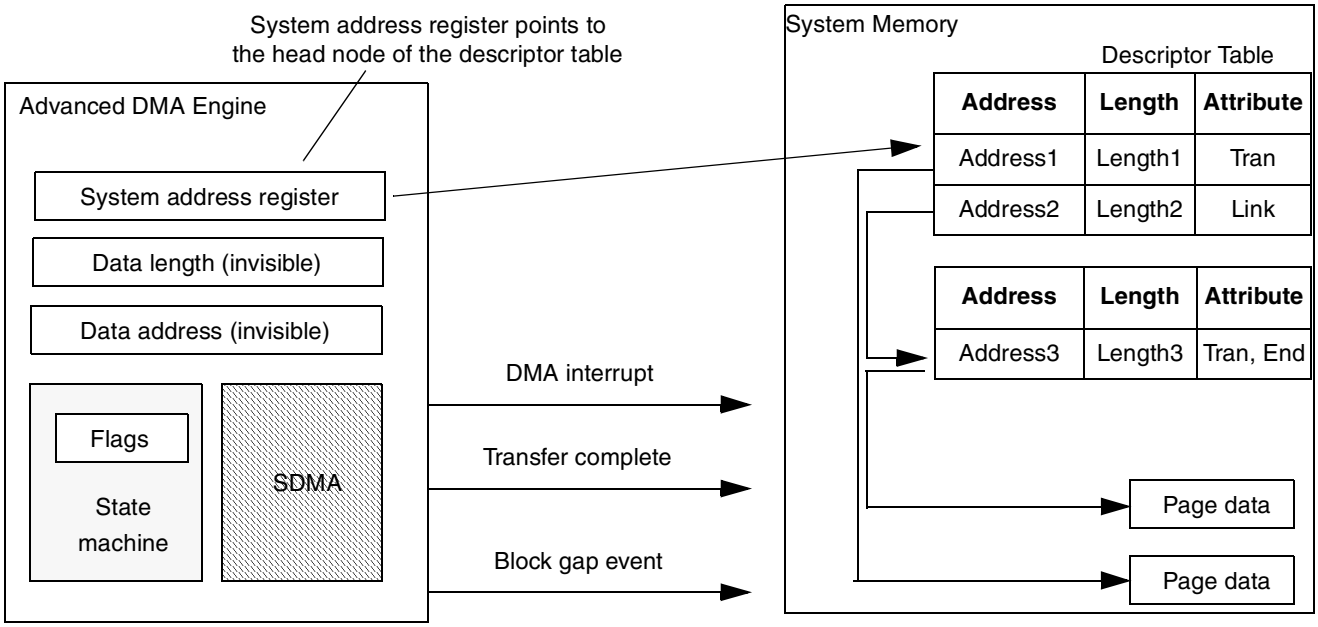


Figure 31-32. ADMA2 Descriptor Table and Access by the ADMA Engine

31.4.2.3.3 ADMA Interrupts

If the Interrupt flag (bit 2) of a descriptor is set, the ADMA engine generates an interrupt according to descriptor type, as follows:

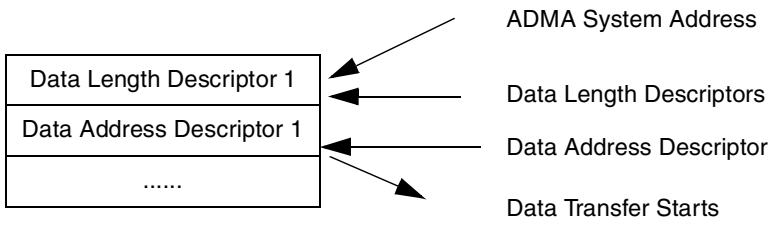
- Nop descriptor: interrupt is generated after the descriptor is fetched
- Rsv descriptor (ADMA2 only): interrupt is generated after the descriptor is fetched
- Set descriptor (ADMA1 only): interrupt is generated after the data length is set
- Tran descriptor: interrupt is generated after this transfer is completed
- Link descriptor: interrupt is generated after the new descriptor address is set.

31.4.2.3.4 ADMA Errors

The ADMA stops execution when the following errors are encountered:

- Descriptor error, generated when the valid flag in the descriptor is cleared. If an ADMA descriptor error occurs, no interrupt is generated, even if the Interrupt flag of this descriptor is set.
- AHB response error
- Data length mismatch error, generated under either of the following conditions:
  - The block count enable bit (BCEN) in the transfer type register is set, and the total data length specified by the descriptor table is different from that specified by the block count and block size.
  - The BCEN bit is cleared, and the total data length is not a multiple of the block size

31.4.3 Register Bank Access Via IP Bus Interface



The IP bus-accessible registers are contained in the register bank. Figure 31-33 is a block diagram of the register bank. Only 32-bit accesses are allowed, and no partial read/writes are supported. All accesses are word aligned (the lowest two address bits are tied to 0).

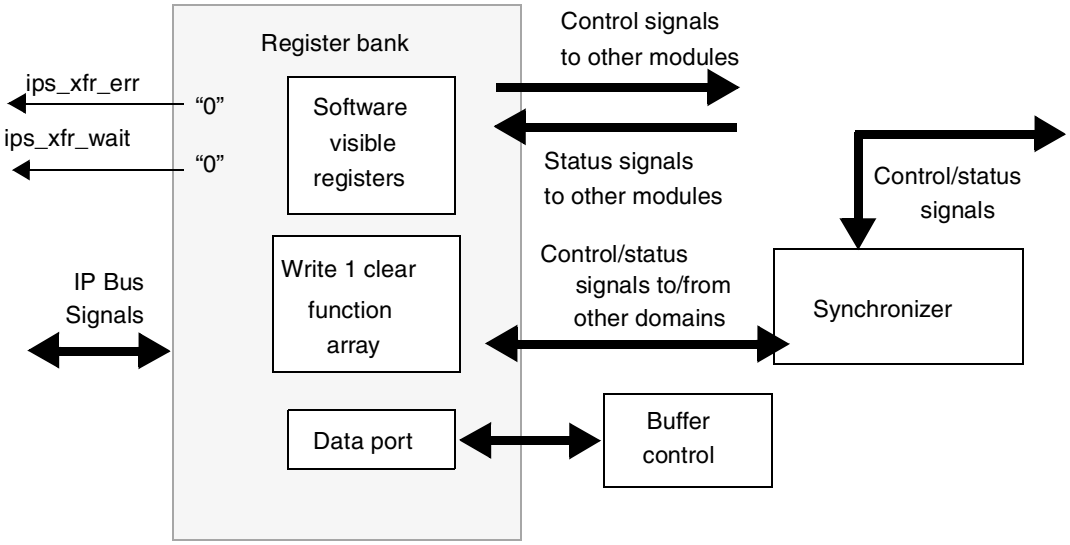


Figure 31-33. Register Bank Diagram

31.4.4 SD Protocol Unit

The SD protocol unit deals with all SD protocol affairs, and performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the DAT[0] line
- Monitors interrupts from the SDIO card
- Asserts the read wait signal
- Gates off the SD clock when the buffer announces danger status
- Detects the write-protect state

The SD protocol unit consists of four submodules:

- SD transceiver
- SD clock and monitor
- Command agent
- Data agent

31.4.4.1 SD Transceiver

The transceiver is the main control submodule in the SD protocol unit. It consists of a finite state machine (FSM) and a control module, from which the control signals for the other three submodules are generated.

31.4.4.2 SD Clock and Monitor

This submodule monitors the signal level on all 8 data lines and the command lines, and directly routes the level values into the register bank. The driver can use these values for debug purposes.

This submodule detects the card detection (CD) line as well as the DAT[3] line. The transceiver reports the card insertion state according to the CD state, or the signal level on the DAT[3] line, when the D3CD bit in the protocol control register is set.

In addition this submodule detects the write protect (WP) line, and informs the register bank when the WP switch is on. When the WP switch is on, the register bank ignores commands involving a write operation.

If the internal data buffer is in danger of overrun or underrun, this submodule asserts the gates off the SD clock. The SD clock is gated on again when the system access of the buffer catches up.

This submodule also drives the LED control output signal (SD_LCTL) when the LCTL bit is set by the driver.

31.4.4.3 Command Agent

The command agent deals with the transactions on the CMD line. Figure 31-34 shows the command CRC shift register.

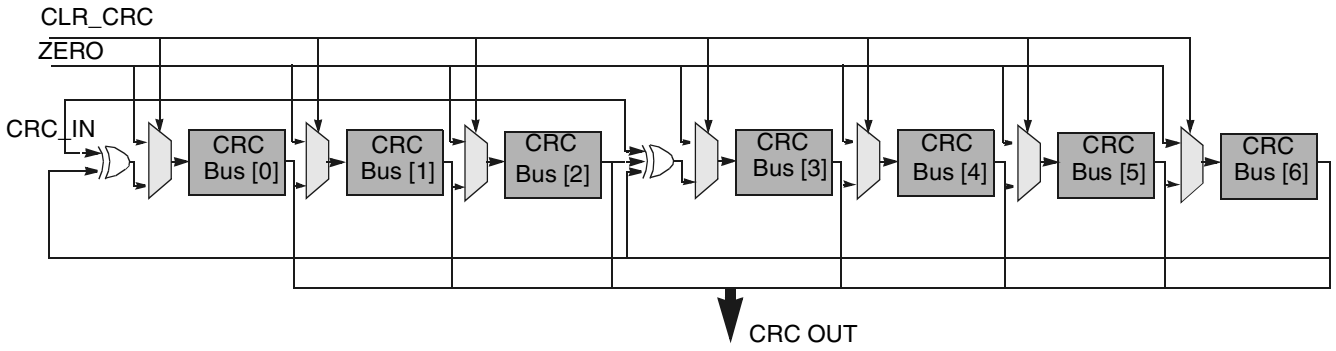


Figure 31-34. Command CRC Shift Register

The CRC polynomial for CMD is computed as follows:

```
Generator polynomial: G(x) = x^7 + x^3 + 1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[6:0] = Remainder [(M(x) * x^7) / G(x)]
```

31.4.4.4 Data Agent

The data agent deals with the transactions on the eight data lines. This module also detects the busy state from the DAT[0] line, and generates the read wait state when requested by the transceiver.

The CRC polynomials for DAT are computed as follows:

```
Generator polynomial: G(x) = x^16 + x^12 + x^5 +1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[15:0] = Remainder [(M(x) * x^16) / G(x)]
```

31.4.5 Clock and Reset Manager Submodule (CRM)

The CRM controls all the reset signals within the eSDHC, which can be classified as four types:

- Hardware reset
- Software reset for all
- Software reset for the data part
- Software reset for the command part

Reset signals are fed into the CRM, and stable signals are generated inside the CRM to reset all other modules.

The CRM also monitors the activities of all other eSDHC submodules, supplies the clocks for them and (when enabled) automatically gates off the corresponding clocks. Clocks used by the eSDHC include IPG clock (ipg_clk), peripheral source clock (ipg_perclk), and master clock (hclk).

31.4.6 SD Clock Generator

The SD clock generator generates the SD clock (SD_CLK) signal from the peripheral source clock by dividing in two stages. Refer to Figure 31-35 for the structure of the divider.

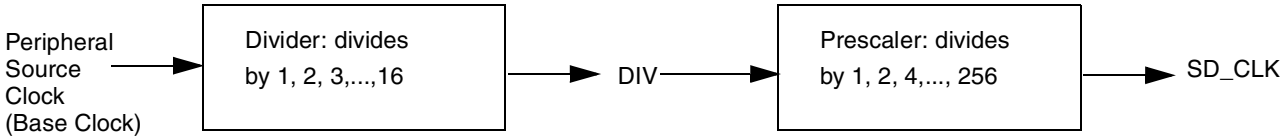


Figure 31-35. Two Stages of the Clock Divider

The SD_CLK signal which is output from the clock divider drives the clocks for all submodules of the SD protocol unit, and for the sync FIFOs to synchronize with the data rate in the internal data buffer (see Figure 31-24).

The divider and prescaler values are determined by the DVS and the SDCLKF bits in the system control register. For example, if the peripheral source clock frequency (ipg_perclk) is 96 MHz and the target SD_CLK frequency is 25 MHz, then choosing SDCLKF as 0x01 (divide by 2) and DVS as

0x1 (divide by 2) yields an SD_CLK frequency of 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to obtain a SD_CLK frequency of 400 kHz, SDCLKF = 0x08 and DVS = 0xE yields the exact clock value of 400 kHz. The highest SD_CLK frequency is equal to the base clock's (peripheral source clock); the second highest frequency is base/2; the lowest frequency is base/4096. The ipg_perclk is about 96 MHz: the reset values of SDCLKFS and DVS correspond to divides of 256 and 1 respectively, so the SD clock frequency after reset is 375 kHz.

According to the SD Physical Specification Version 1.1 and the SDIO Card Specification Version 1.2, the maximum SD clock frequency is 50 MHz and must never exceed this limit.

If the peripheral source clock has duty cycle of 50% (which is usually true), then the duty cycle of SD_CLK is also 50%.

31.4.7 SDIO Card Interrupts

This section discusses SDIO card interrupts in the following modes: 1 bit and 4 bit.

31.4.7.1 Interrupts in 1-Bit Mode

In 1-bit mode the DAT[1] signal is dedicated to providing the interrupt function. An interrupt is asserted by pulling DAT[1] low from the SDIO card, until the interrupt service routine clears the interrupt.

31.4.7.2 Interrupts in 4-Bit Mode

In 4-bit mode the interrupt and data line 1 share pin 8. To accommodate this sharing, the eSDHC treats pin 8 as an interrupt only during specified time intervals known as interrupt periods. At all other times, the level on pin 8, is treated as a data signal.

The definition of the interrupt period is different for single-block and multi-block data transfers:

- For single-block transfers, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until the card has received the end bit of the next command that has a data block transfer associated with it.
- For multiblock data transfers, the interrupt period is limited to two clock cycles, due to the limited time between blocks. The interrupt period begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DAT[1] line is held low for one clock cycle with the last clock cycle pulling DAT[1] high. On completion of the interrupt period, the card releases the DAT[1] line into the high-Z state. The eSDHC samples the DAT[1] during the interrupt period when the IABG bit in the protocol control register is set.

Refer to SDIO Card Specification v1.10f for further information about SDIO card interrupts.

31.4.7.3 Card Interrupt Handling

Before servicing a card interrupt, the host driver is responsible to clear the card interrupt interrupt enable (CINTIEN) bit in the interrupt signal enable register. This prevents inadvertent interrupts from occurring while the interrupt is being serviced. After all interrupt requests from the card are cleared, it is the host driver's responsibility to reset this bit to 1.

The SDIO status bit is cleared by resetting the SDIO interrupt. Writing to this bit would have no effects. In 1-bit mode, the eSDHC will detect the SDIO interrupt with or without the SD clock (to support wakeup). In 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the Host System Interrupt Controller. When the SDIO status has been set, and the host driver needs to start the interrupt service routine, the SDIO bit in the Interrupt Control Register of SDIO card will be cleared. This is required to clear the SDIO interrupt status latched in the eSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO interrupt enable bit is set to 1, and the eSDHC starts sampling the interrupt signal again.

Figure 31-36 shows the system's interrupt-handling features, and also shows a flowchart that summarizes the interrupt detection and handling procedure.

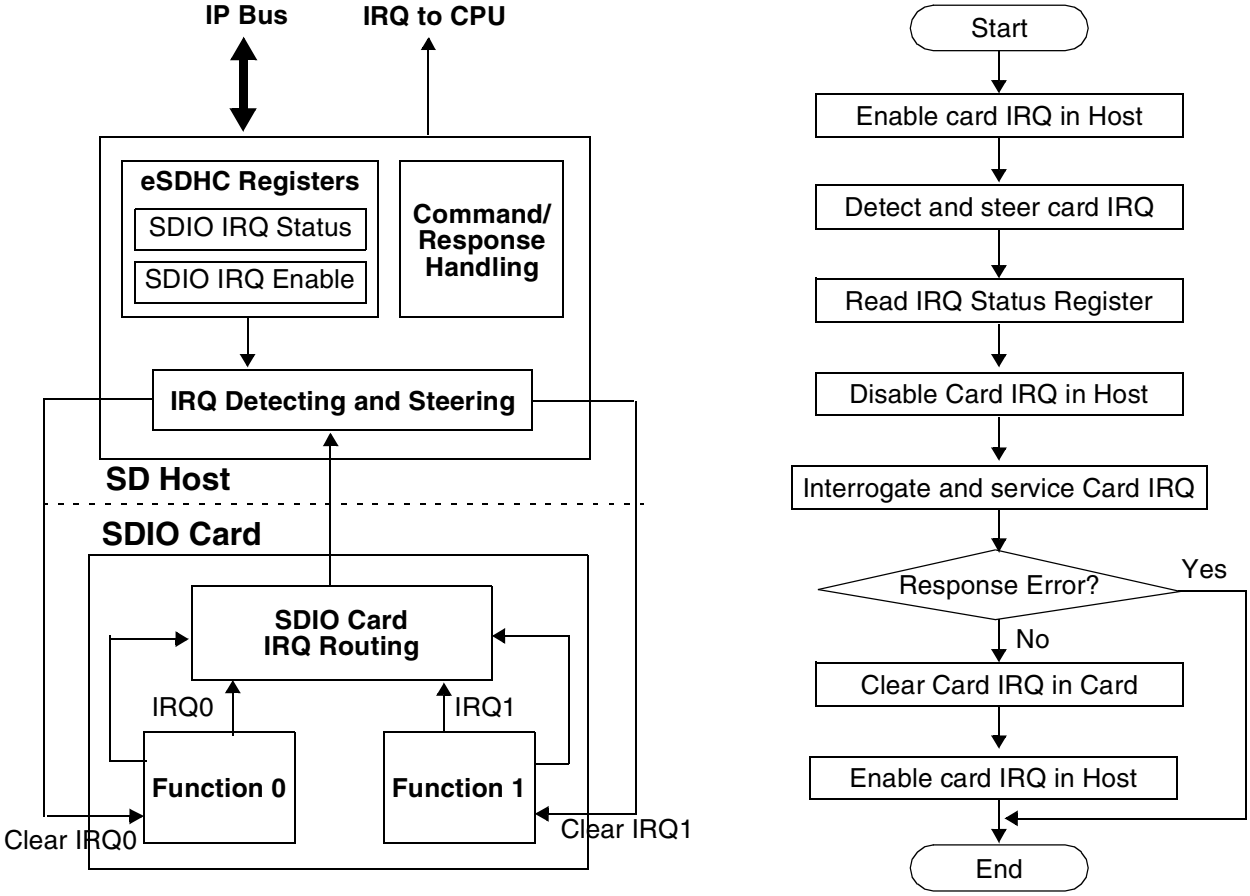


Figure 31-36. Card Interrupt-Handling Features and Card Interrupt-Detection and Handling Procedure

31.4.8 Card Insertion and Removal Detection

The eSDHC uses either the DAT[3] or the SD_CD signal to detect card insertion or removal. When there is no card on the MMC/SD bus, the DAT[3] is pulled to a low voltage level by default. When any card is inserted to or removed from the socket, the eSDHC detects the logic value changes on the DAT[3] pin and

generates an interrupt. Whether DAT[3] signal is used for card detection or not, the SD_CD pin must be connected for card detection. It may be implemented by a GPIO. The SD_CD pin is always used as a reference for card detection. If either the DAT[3] or SD_CD signal reports card inserted, the eSDHC informs the host system that a card is inserted and an interrupt is sent if it is enabled.

31.4.9 Power Management and Wakeup Events

The eSDHC offers a power management feature: by clearing the PEREN, HCKEN or IPGEN bits in the system control register, the peripheral source clock, master clock, or IPG clocks are gated in the low position to the eSDHC. For maximum power saving, the user can disable all the clocks to the eSDHC when no operation is in progress.

Before the software disables the host clock, it should ensure that all of the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- The internal data buffer is empty

Even when the system is in low power mode and the clocks to the eSDHC are disabled, there are some events for which the clocks must be enabled to handle the event. There are three events (denoted as wakeup events or wakeup interrupts) that can be used to wake up the eSDHC by re-enabling the clocks, even if there are no clocks enabled. These three wakeup interrupts are:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from the SDIO card

In order to enable these interrupts to wake up the eSDHC, the corresponding wakeup enabled bits need to be set in the protocol control register, before the CPU enters sleep mode. See Section 31.3.3.8, Protocol Control Register (PROCTL)” for more details.

31.5 Initialization/Application Information

All communication between system and cards is controlled by the host. See Section 31.6, MMC/SD/SDIO/CE-ATA Card Commands for a complete list of the commands that the host can issue. There are two types of commands:

- Broadcast commands are intended for all cards: examples include “GO_IDLE_STATE”, “SEND_OP_COND”, “ALL_SEND_CID” and so on. In broadcast mode, all cards are in open-drain mode to avoid bus contention.
- Addressed commands can be issued after the broadcast command CMD3 has been sent, causing the cards to enter standby mode. In standby mode, the CMD and DAT I/O pads are in push-pull mode, which provides the driving capability for maximum frequency operation.

31.5.1 Command Send and Response Receive Basic Operation

The following pseudocode indicates the procedure for sending a command to the card(s). The data type WORD here denotes unsigned 32-bit integer.

Example 31-1. Pseudocode for Sending a Command to the Card(s)

```
send_command(cmd_index, cmd_arg, other requirements)
{
WORD wCmd; // 32-bit integer to make up the data to write into transfer type register, it is
recommended to implement in a bit-field manner
wCmd = (<cmd_index> & 0x3f) >> 24; // set the first 8 bits as ‘00’+<cmd_index>
set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL accorind to the command index;
if (internal DMA is used) wCmd |= 0x1;
if (multi-block transfer) {
         set MSBSEL bit;
         if (finite block number) {
                  set BCEN bit;
                  if (auto12 command is to use) set AC12EN bit;
         }
}
write_reg(CMDARG, <cmd_arg>); // configure the command argument
write_reg(XFERTYP, wCmd); // set transfer type register as wCmd value to issue the command
}
wait_for_response(cmd_index)
{
while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
read IRQ Status register and check if any error bits are set
if (any error bits are set) report error;
write 1 to clear CC bit and all Command Error bits;
}
```

In this example, the function wait_for_response is implemented by means of polling for the sake of simplicity. For an effective and formal way, the response is checked after the command complete interrupt is received. By doing this, the corresponding interrupt status bits are verified.

In some scenarios, a response timeout is expected after a command is issued. For example, after the host issues a CMD3 command and all cards enter the standby state, then when CMD2 is sent the cards send no response to the host. The host driver is responsible to deal with these ‘fake’ errors.

31.5.2 Card Identification Mode

When a card is inserted into the socket, or when the card is reset by the host, the host is responsible for performing a sequence of operations on the card including the following:

- Card detection (when the card is inserted) or card reset (when the card is reset by the host)
- Voltage validation (also determines whether the card is MMC, SD, SDIO, or CE-ATA)
- Card registration (including card identification and determination of relative card address (RCA))

31.5.2.1 Card Detection

Figure 31-37 shows a flowchart for the detection of MMC, SD and SDIO cards using the eSDHC.

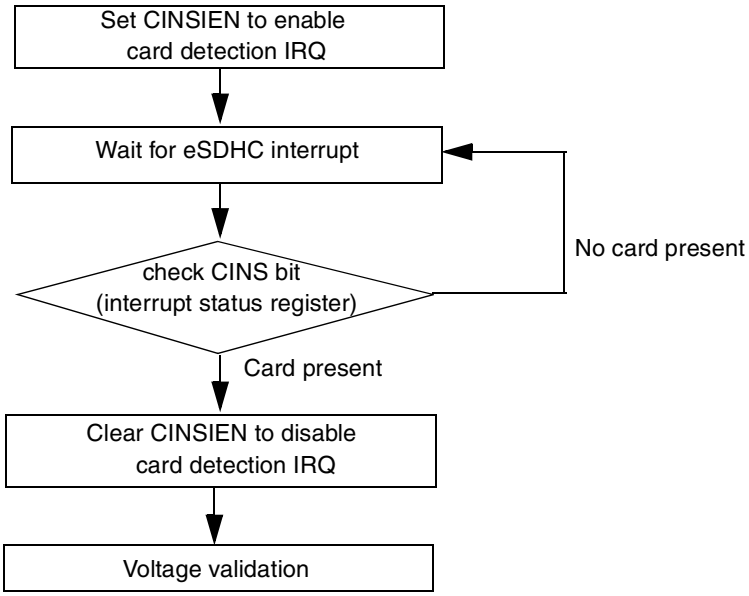


Figure 31-37. Flow Diagram for Card Detection

31.5.2.2 Reset

There are three types of resets involving the eSDHC:

- Hardware reset (card and host), which is driven by POR (power on reset)
- Software reset (host only). There are three types of software reset, which are effected by writes to bits in the system controller register as follows:
  - — Setting the RSTD bit to 1 resets the data part of the eSDHC
  - — Setting the RSTC bit to 1 resets the command part of the eSDHC
  - — Setting the RSTA bit to 1 resets all parts of the eSDHC.
- Card reset (Card Only). The command, “Go_Idle_State” (CMD0), is the software reset command for all types of MMC, SD Memory, and CE-ATA cards. This command sets each card into the idle state regardless of the current card state. For SD I/O Cards, CMD52 is used to write an I/O reset in the CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host is responsible to validate the voltage range of the card. See Figure 31-38 for a pseudocode to reset both the eSDHC and the card.

For CE-ATA devices that support ATA mode, two CMD39 commands should be issued back-to-back to the ATA control register before issuing CMD0 to reset the MMC layer. The first CMD39 has the SRST bit set to one, and the second has the SRST bit cleared.

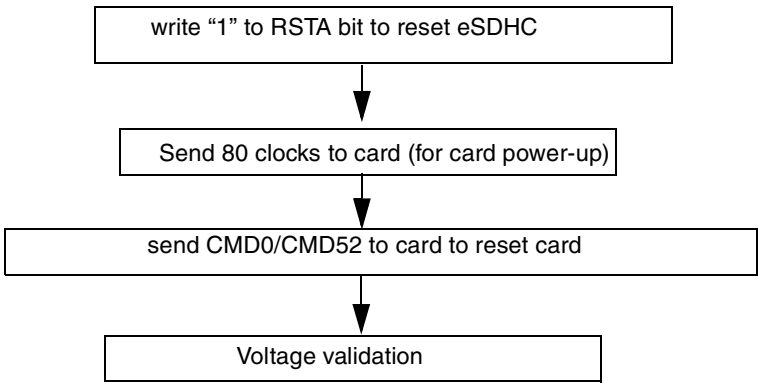


Figure 31-38. Flowchart for Reset of the eSDHC and SD I/O Card

Example 31-2. Pseudocode for Resetting eSDHC and Card

```
software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the host
set DTOCV and SDCLKFS bit fields to get the SD_CLK of frequency around 400kHz
configure IO pad to set the power voltage of external card to around 3.0V
poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
```

31.5.2.3 Voltage Validation

The voltage validation procedure is designed to provide a mechanism to identify and reject cards which do not match the Vdd range(s) desired by the host.

A card’s supported minimum and maximum values for Vdd are defined in the card’s operating conditions register (OCR).The supported range may not cover the maximum allowed voltage range specified in the card specification. Cards that store the card identification (CID) and card-specific data (CSD) in preloaded memory are only able to communicate this information under data transfer Vdd conditions: so if the host and card have non-overlapping Vdd ranges, then the card is not able to complete the identification cycle, nor is it be able to send CSD data.

Voltage validation makes use of the special commands listed in Table 31-36 as follows:

- The host can send a command in Table 31-36 with the desired Vdd voltage window as operand. Then cards of the corresponding type that cannot perform the data transfer in the specified range disqualify themselves from further bus operations, and go into the inactive state.
- Alternatively, the host can send a command in Table 31-36 and omits the voltage range in the command. By this means the host can query each card and determine the common voltage range before sending out-of-range cards into the inactive state. This query can be used to enable the host

to select a common voltage range, or to enable notification of the system when an unusable card is detected in the stack.

Table 31-36. Voltage Validation Commands for Different Card Types

Card Type	Command Name	CMD #
MMC, CE-ATA	Send_Op_Cont	CMD1
SD	SD_Send_Op_Cont	ACMD41
SDIO	IO_Send_Op_Cont	CMD5

The following pseudocode outlines the voltage validation procedure when a card is inserted:

Example 31-3. Pseudocode for Voltage Validation Procedure Upon Card Insertion

```
voltage_validation(voltage_range_argument)
{
label the card as UNKNOWN;
send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
operating voltage, command argument is zero
if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
        if (0 < number of IO functions) {
                label the card as SDIO;
                IORDY = 0;
                while (!(IORDY in IO OCR response)) { // set voltage range for each IO function
                        send_command(IO_SEND_OP_COND, <voltage range>, <other parameter>);
                        wait_for_response(IO_SEND_OP_COND);
                } // end of while ...
        } // end of if (0 < ...
        if (memory part is present inside SDIO card) Label the card as SDCombo; // this is an
SD-Combo card
} // end of if (RESP_TIMEOUT ...
if (the card is labelled as SDIO card) return; // card type is identified and voltage range is
set, so exit the function;
send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific CMD
prefix
if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
        send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage range
for memory part or SD card
        wait_for_response(SD_APP_OP_COND); // voltage range is set
        if (card type is UNKNOWN) label the card as SD;
        return; //
} // end of if (no error ...
else if (errors other than time-out occur) { // command/response pair is corrupted
        deal with it by program specific manner;
} // of else if (response time-out
else { // CMD55 is refused, it must be MMC card or CE-ATA card
        if (card is already labelled as SDCombo) { // change label
                re-label the card as SDIO;
                ignore the error or report it;
                return; // card is identified as SDIO card
        } // of if (card is ...
        send_command(SEND_OP_COND, <voltage range>, <...>);
        if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted, either
                label the card as UNKNOWN;
```

```
                return;
        } // of if (RESP_TIMEOUT ...
        if (check for CE-ATA signature succeeded) { // the card is CE-ATA
                store CE-ATA specific info from the signature;
                label the card as CE-ATA;
        } // of if (check for CE-ATA ...
        else label the card as MMC;
} // of else
}
```

The host detects a CE-ATA device by issuing a CMD 60 to check for a CE-ATA signature, after completing all other MMC voltage validation and identification steps. If the device responds to CMD 60 with the CE-ATA signature, then a CE-ATA device has been found.

It is possible that the CE-ATA device does not support the ATA mode, so the driver cannot issue ATA command to the device. In order to check this, the driver queries the EXT_CSD register byte 504 (S_CMD_SET) in the MMC register space. If the ATA bit (bit 4) is set, then ATA commands are supported, and the driver sets the ATA bit (bit 4) of the EXT_CSD register byte 191 (CMD_SET) to activate the ATA command set for use. To choose the ATA command set, the driver issues CMD6.

31.5.2.4 Card Identification and Registration

The card identification and registration process establishes a relative card address (RCA), which is used to address the card for future data transfer operations. The registration processes for different card types are described in the following sections:

- Section 31.5.2.4.1, Card Identification and Registration for MMC Cards”
- Section 31.5.2.4.2, Card Registration for SD, SDIO, and SD Combo Cards”
- Section 31.5.2.4.3, Card Identification and Registration for CE-ATA Cards”

A pseudocode that outlines the entire registration process is given in Section 31.5.2.4.4, Card Registration Pseudocode.”

31.5.2.4.1 Card Identification and Registration for MMC Cards

For MMC cards, the relative card address (RCA) is set by the host. The procedure for identifying the cards and setting their RCAs is described below.

Card identification mode for MMC cards uses a clock frequency of less than 400 kHz and a power voltage greater than 2.7 V. The process begins in open-drain mode: the open-drain driver stages on the CMD line allow parallel card operation during card identification. The host sends the following sequence of commands:

1. After the bus is activated, the host broadcasts a CMD1 command requesting the cards to send their valid operating conditions. The response to CMD1 is the “wired OR” operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the inactive state.
2. The host then broadcasts an All_Send_CID (CMD2) command, asking all cards for their unique card identification (CID) number. All unidentified cards (in ready state) simultaneously start sending their CID numbers serially, while bitwise monitoring their outgoing bit stream. Those

cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods stop sending their CID immediately, and must wait for the next identification cycle. Since the CID is unique for each card, only one card can successfully send its full CID to the host. This card then goes into the identification state.

3. Next, the host issues a point-to-point Set_Relative_Addr command (CMD3) to assign to the card a relative card address (RCA). Once the RCA is received, the card state changes to the standby state, and the card does not participate in further identification cycles. The card's output driver switches from open-drain to push-pull.
4. The host repeats steps 2 and 3 above until the host sends a CMD2 which times out, signifying the completion of the identification process.

31.5.2.4.2 Card Registration for SD, SDIO, and SD Combo Cards

For SD, SDIO, and SD Combo cards, the RCA is published by the cards at the host's request. The procedure for identifying the cards and setting their RCAs is described below.

Card identification mode for SD cards uses a clock frequency of less than 400 kHz and a power voltage greater than 2.7 V (as defined in the SD card specification). The CMD line output drivers are in push-pull mode. The host sends the following sequence of commands:

1. After the bus is activated, the host broadcasts a ACMD41 (for SD) or CMD5 (for SDIO) command to all new cards in the system, requesting them to send their valid operating conditions. The card responds by sending the contents of its operating conditions register. Incompatible cards are put into the inactive state.
2. The host then broadcasts an All_Send_CID (CMD2) command, asking all cards for their unique card identification (CID) number. All unidentified cards (in ready state) simultaneously start sending their CID numbers serially, while bitwise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods stop sending their CID immediately, and must wait for the next identification cycle. Since the CID is unique for each card, only one card can successfully send its full CID to the host. This card then goes into the identification state.
3. Next, the host issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. Once the RCA is received, the card changes its state to the standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.
4. The host repeats steps 2 and 3 above until the host sends a CMD2 which times out, signifying the completion of the identification process.

31.5.2.4.3 Card Identification and Registration for CE-ATA Cards

CE-ATA operation has the same interface as MMC cards. However, CE-ATA devices are connected in a point-to-point manner, and no RCA is needed. All data communications in card identification mode use the command line (CMD) only. See CE-ATA Digital Protocol, Revision 1.1 for more details.

CE-ATA enters the tran state after reset is completed.

31.5.2.4.4 Card Registration Pseudocode

The following pseudocode outlines the card registration process for all cards (corresponding to steps 2 through 4 in the procedures shown in Section 31.5.2.4.1, Card Identification and Registration for MMC Cards" and Section 31.5.2.4.2, Card Registration for SD, SDIO, and SD Combo Cards").

Example 31-4. Pseudocode for Card Registration Process—All Cards

```
card_registry()
{
do { // decide RCA for each card until response time-out
        if(card is labelled as SDCombo or SDIO) { // for SDIO card like device
                send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to publish its
RCA
                retrieve RCA from response;
        } // end if (card is labelled as SDCombo ...
        else if (card is labelled as SD) { // for SD card
                send_command(ALL_SEND_CID, <...>);
                if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
                send_command(SET_RELATIVE_ADDR, <...>);
                retrieve RCA from response;
        } // else if (card is labelled as SD ...
        else if (card is labelled as MMC or CE-ATA) { // treat CE-ATA as MMC
                send_command(ALL_SEND_CID, <...>);
                rca = 0x1; // arbitrarily set RCA, 1 here for example, this RCA is also the
relative address to access the CE-ATA card
                send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper 16
bits
        } // end of else if (card is labelled as MMC ...
} while (response is not time-out);
}
```

31.5.3 Card Accesses

This section describes write, read, suspend/resume, and ADMA card accesses.

31.5.3.1 Block Write

Normal writes and writes with pause are described in Section 31.5.3.1.1, Normal Write" and Section 31.5.3.1.2, Write with Pause," respectively.

31.5.3.1.1 Normal Write

During a block write (CMD24–27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates the failure on the DAT line. The transferred data is discarded and not written, and all further transmitted blocks (in multi-block write mode) is ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set, and the CE-ATA card does not support partial block write, either), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the

status register, while ignoring all further data transfer, waits in the Receive-data-State for a stop command. For a CE-ATA card, check the CE-ATA card specification for its behavior in block misalignment. The write operation is also aborted if the host tries to write over a write protected area.

For MMC and SD cards, programming of the CID and CSD registers does not require a previous block size setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and not change any register contents.

Different cards may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DAT line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO and CE-ATA cards at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby State and release the DAT line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling DAT to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
   a) For SD/MMC cards, use SET_BLOCKLEN (CMD16)
   b) For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1–7)
   c) For CE-ATA cards, configure bits 1~0 in the scrControl register
3. Set the eSDHC block size register to be the same as the block size set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the transfer complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

31.5.3.1.2 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the SD_CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks. As there is no time-out

condition during a write operation to the data blocks, a write to all types of cards can be paused in this way, and if the DAT0 line is not required to negate to release the busy state, no suspend command is needed.

Like in the flow described in Section 31.5.3.1.1, Normal Write," the write with pause is shown with the same kind of write operation:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
   a) For SD/MMC, use SET_BLOCKLEN (CMD16)
   b) For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1–7)
   c) For CE-ATA cards, configure bits 1:0 in the scrControl register
3. Set the eSDHC block size register to be the same as the block size set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the transfer complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the transfer complete interrupt.
12. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. The data transfer and the setting of the SABGREQ bit are concurrent, and the delay of the register read and writing, the actual number of blocks left may not be exactly the value read earlier. The driver reads the value of BLKCNT after the transfer is paused and the transfer complete interrupt is received.

It is also possible the last block has begun when the stop at block gap request is sent to the buffer. In this case, the next block gap is actually the end of the transfer. These types of requests are ignored and the driver should treat this as a non-pause transfer and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the Host System is not stopped, and the transfer is active until the data buffer is full. Because of this (if not needed), it is recommended to avoid using the Suspend Command for the SDIO card. This is because when such a command is sent, the eSDHC thinks the System switches to another function on the SDIO card, and flush the data buffer. The eSDHC takes the Resume Command as a normal command with data transfer, and it is left for the driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the transfer type register are set as well as the AC12EN bit. However, the eSDHC automatically sends a CMD12 to mark the end of the multi-block transfer.

31.5.3.2 Block Read

Normal reads and reads with pause are described in Section 31.5.3.2.1, Normal Read" and Section 31.5.3.2.2, Read with Pause," respectively.

31.5.3.2.1 Normal Read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer State. For multi-block read, data blocks is continuously transferred until a stop command is issued.

The software flow to read from a card incorporates the internal DMA, and the read operation is a multi-block read with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status) with different transfers, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
   a) For SD/MMC, use SET_BLOCKLEN (CMD16)
   b) For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1–7)
   c) For CE-ATA cards, configure bits 1:0 in the scrControl register
3. Set the eSDHC block size register to be the same as the block size set for the card in Step 2.
4. Set the eSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer read ready interrupt, configure the DMA settings and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set:
6. Wait for the transfer complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or some another error, occurred during the auto12 command sending and response receiving.

31.5.3.2.2 Read with Pause

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the read wait feature can pause during the read operation. If the SDIO card supports read wait (SRW bit in CCCR register is 1), the driver can set the SABGREQ bit in the protocol control register to pause the transfer between the data blocks. Before setting the SABGREQ bit, make sure the RWCTL bit in the protocol control register is set, otherwise the eSDHC does not assert the read wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit once the Read Wait capability of the SDIO card is recognized.

As in the flow described in Section 31.5.3.2.1, Normal Read," the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm the card supports read wait.
2. Set the RWCTL bit.
3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
   a) For SD/MMC, use SET_BLOCKLEN (CMD16)
   b) For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1–7)
   c) For CE-ATA cards, configure bits 1~0 in the scrControl register
5. Set the eSDHC block size register to be the same as the block size set for the card in Step 2.
6. Set the eSDHC number block register (NOB), nob is 5 (for instance).
7. Disable the buffer read ready interrupt, configure the DMA setting and enable the eSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set
8. Set the SABGREQ bit.
9. Wait for the transfer complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the transfer complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or some another error, occurred during the auto12 command sending and response receiving.

Like the write operation, it is possible to meet the ending block of the transfer when paused. In this case, the eSDHC ignores the stop at block gap request and treat it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. Whether the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, the eSDHC takes the command as a normal one accompanied with data transfer. It is left for the driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the transfer type register are set, as well as the AC12EN bit. However, the eSDHC automatically sends the CMD12 to mark the end of multi-block transfer.

31.5.3.3 Suspend/Resume Operations

The eSDHC supports suspend and resume operations for SDIO cards, although the implementation of suspend is slightly different than that suggested in the SDIO Card Specification.

31.5.3.3.1 Suspend Operation

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of the SDIO card. The eSDHC does not monitor the content of the response, so it doesn't know if the Suspend command succeeded or not. Accordingly, it does not negate read wait for a read pause. To solve this problem, the driver first sends the command as if it were a normal command (CMDTYP = 01). After the command succeeds, and the BS bit is set to 1 in the response, the driver then sends another command marked as Suspend to inform the eSDHC that the current transfer is suspended. The following sequence is used for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field is set as 0b00.
3. Check the BS bit of the CCCR in the response. If it is set, repeat this step until the BS bit is cleared or abandon the Suspend operation according to the driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 0b01, to inform the eSDHC that the paused operation has successfully suspended. If the paused transfer is a read operation, the eSDHC stops driving DAT2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA system address register (for internal DMA operation), and the block attribute register.
6. Begin operation for another function on the SDIO card.

31.5.3.3.2 Resume Operation

Data transfer is resumed following a Suspend command according to the following procedure:

1. Resume the suspended function by restoring the context register with the value saved in step #5 of the Suspend operation (see Section 31.5.3.3.1, Suspend Operation").
2. Send the Resume command. In the transfer type register, all bit fields are set to the value as if this were another ordinary data transfer, instead of a transfer resume (except the CMDTYP field is set to 0b10).
3. If the Resume command has responded, the data transfer is resumed.

31.5.3.4 ADMA Usage

To use the ADMA in a data transfer, the host driver prepares the correct descriptor chain prior to sending the read/write command. The steps to accomplish this are:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length must be a multiple of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4 KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1–3.
4. Repeat steps 1–3 until all descriptors are created.

5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors match the product of the block size and block number configured in the block attribute register.
6. Set the ADMA system address register to the address of the first descriptor and set the DMAS field in the protocol control register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the transfer type register.

Steps 1–5 are independent of step 6, so step 6 can finish before steps 1–5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

31.5.3.5 Handling Transfer Errors

This section discusses the following transfer errors: CRC, internal DMA, ADMA, and Auto CMD12.

31.5.3.5.1 CRC Error

It is possible that at the end of a block transfer, a write CRC status error or read CRC error occurs. For this type of error the last block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one. For a multi-block transfer, the host driver issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even if AC12EN and BCEND bits are set, the eSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by the eSDHC. In this case, the driver resends or re-obtains the last block with a single-block transfer.

31.5.3.5.2 Internal DMA Error

During the data transfer with internal simple DMA, if the DMA engine encounters some error on the AHB bus, the DMA operation is aborted and DMA error interrupt is sent to the Host System. When acknowledged by such an interrupt, the driver calculates the start address of data block in which the error occurs. The start address can be calculated by either of the following methods:

1. Read the DMA system address register. The error occurs during the previous burst. Taking the block size, the previous burst length and the start address of the next burst transfer into account, it is straightforward to obtain the start address of the corrupted block.
2. Read the BLKCNT field of the block attribute register. By the number of blocks left, the total number of blocks to transfer, the start address of transfer, and the size of each block, the start address of the corrupted block can be obtained. When the BCEN bit is not set, the contents of the block attribute register do not change, so this method does not work.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

31.5.3.5.3 ADMA Error

There are three types of ADMA errors: AHB transfer, invalid descriptor, and data length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set. For

acknowledging the status, the host driver should recover the error as shown below and retransfer from the place of interruption.

- AHB transfer error

  Such errors can occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and retransfer the block that was corrupted, or the next block if no block is corrupted.
- Invalid descriptor error

  For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data length mismatch error

  Recovery from data length mismatch is similar to recovery from an invalid descriptor error. The host driver polls relating registers to retrieve the transfer context, applies a reset for the data part, configures a new descriptor chain, and makes another transfer if there is data left. As in the case of invalid descriptor error, the data length must match the new transfer.

31.5.3.5.4 Auto CMD12 Error

If the AC12EN bit is set when a multi-block data transfer is initiated by the data command, then the eSDHC automatically sends a CMD12 to the card to stop the transfer after the last block of the transfer is sent or received. It is recommended that the driver respond to errors in the auto CMD12 command as follows:

1. If the error is an auto CMD12 response timeout (indicated by bit 1 of the auto CMD12 error status register (AUTOC12ERR), then it is not certain whether the command has been accepted by the card or not. In this case, the driver can clear the AUTOC12ERR status bits and resend the CMD12 until it is accepted by the card.
2. Auto CMD12 response CRC error (indicated by bit 3 of AUTOC12ERR). Since the card responds to CMD12, the card aborts the transfer. In this case, the driver can ignore the error and clear the error status bit.
3. Auto CMD12 conflict error or not sent. The CMD12 command s not sent. In this case, the driver can send a CMD12 manually.

31.5.3.6 Card Interrupts

The external cards can inform the host controller by means of some special signals. For the SDIO card, it can be the low level on the DAT[1] line during some special period. For the CE-ATA card, it can be a pulse on the CMD line to inform the host controller that the command and its response is finished, and it is possible that some additional external interrupt behaviors are defined. The eSDHC only monitors the DAT[1] line and supports the SDIO interrupt.

When the SDIO interrupt is captured by the eSDHC, and the host system is informed by the eSDHC asserting the eSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt factor is controlled by the external card, the interrupt from the SDIO card must be serviced before the CINT bit is cleared by writing 1. Refer to Section 31.4.7.3, Card Interrupt Handling" for the card interrupt handling flow.

31.5.4 Switch Function

MMC cards transferring data at bus widths other than 1-bit is a new feature added to the MMC spec. High-speed timing mode for all card devices, is also a recent addition to the various card specifications. To enable these new features in cards of different types the host driver issues commands as follows:

- For MMC cards (and CE-ATA over MMC interface), the high-speed mode is queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol SWITCH). 4-bit and 8-bit MMC bus widths are also enabled by SWITCH commands, but with a different argument.
- For SD cards, the high-speed mode is queried and enabled by a CMD6 (with the mnemonic symbol SWITCH_FUNC).
- For SDIO cards, the high-speed mode is enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed.

These new functions can be disabled by a software reset. For SDIO cards this can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

The following subsections provide pseudocodes for enabling/disabling high-speed mode for various card types and for setting MMC bus width. For the sake of simplicity, the pseudocodes do not show current capability check, which is recommended in the function switch process.

31.5.4.1 Query, Enable, and Disable SDIO High-Speed Mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is ‘0’) report the SDIO card does not support high-speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is cleared;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

31.5.4.2 Query, Enable and Disable SD High-Speed Mode

```
enable_sd_high_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
```

```
send CMD6, with argument 0xFFFFF1 and read 64 bytes of data accompanying the R1 response;
wait data transfer done bit is set;
check if the bit 401 of received 512 bit is set;
if (bit 401 is ‘0’) report the SD card does not support high-speed mode and return;
send CMD6, with argument 0x80FFFFF1 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sd_high_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

31.5.4.3 Query, Enable and Disable MMC High-Speed Mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high-speed mode and return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the ‘high-speed mode’ in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high-speed mode failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into eSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

31.5.4.4 Set MMC Bus Width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and return;
send CMD6 with argument 0x3B70x00; (8-bit, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

31.5.5 ADMA Operation

Pseudocodes for ADMA1 and ADMA2 operation are provided in Section 31.5.5.1, ADMA1 Operation" and Section 31.5.5.2, ADMA2 Operation" respectively.

31.5.5.1 ADMA1 Operation

```
Set_adma1_descriptor
{
      if (to start data transfer) {
          // Make sure the address is 4KB align.
          Set ‘Set’ type descriptor;
          {
               Set Act bits to 01;
               Set [31:12] bits data length (byte unit);
          }
          Set ‘Tran’ type descriptor;
          {
               Set Act bits to 10;
               Set [31:12] bits address (4KB align);
          }
      }
      else if (to fetch descriptor at non-continuous address) {
          Set Act bits to 11;
          Set [31:12] bits the next descriptor address (4KB align);
      }
      else { // other types of descriptor
          Set Act bits accordingly
      }
      if (this descriptor is the last one) {
          Set End bit to 1;
      }
      if (to generate interrupt for this descriptor) {
          Set Int bit to 1;
      }
      Set Valid bit to 1;
}
```

31.5.5.2 ADMA2 Operation

```
Set_adma2_descriptor
{
      if (to start data transfer) {
          // Make sure the address is 32-bit boundary (lower 2-bit are always '00').
```

```
            Set higher 32-bit of descriptor for this data transfer initial address;
            Set [31:16] bits data length (byte unit);
            Set Act bits to '10';
        }
        else if (to fetch descriptor at non-continuous address) {
            Set Act bits to '11';
            // Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
            Set higher 32-bit of descriptor for the next descriptor address;
        }
        else { // other types of descriptor
            Set Act bits accordingly
        }
        if (this descriptor is the last one) {
            Set 'End' bit '1';
        }
        if (to generate interrupt for this descriptor) {
            Set 'Int' bit '1';
        }
        Set the 'Valid' bit to '1';
}
```

31.6 MMC/SD/SDIO/CE-ATA Card Commands

There are four kinds of commands defined to control multimedia cards:

- Broadcast commands (bc), which elicit no response from the cards
- Broadcast commands with response (bcr), which elicit responses from all cards simultaneously
- Addressed (point-to-point) commands (ac), which do not involve data transfer on the DAT lines
- Addressed (point-to-point) data transfer commands (adtc), which involve data transfer

Table 31-37 lists commands for MMC/SD/SDIO/CE-ATA cards. Refer to the corresponding specifications for more details about these commands.

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards

CMD INDEX	Type	Argument	Response	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	—	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	—	SET_DSR	Programs the DSR of all cards.

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards (continued)

CMD INDEX	Type	Argument	Response	Abbreviation	Description
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send their operating conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to “SD Physical Specification V1.1” for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to “The MultiMediaCard System Specification Version 4.0 Final draft 2” for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/DESELECT_CARD	Toggles a card between the standby and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards (continued)

CMD INDEX	Type	Argument	Response	Abbreviation	Description
CMD15	ac	[31:6] RCA [15:0] stuff bits	—	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	Reserved				
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.
CMD21-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards (continued)

CMD INDEX	Type	Argument	Response	Abbreviation	Description
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards (continued)

CMD INDEX	Type	Argument	Response	Abbreviation	Description
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather that a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57~59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	CE-ATA devices contain a set of Status and Control registers that begin at register offset 80h. These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer for the ATA command.
CMD62~63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width (‘00’=1bit or ‘10’=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.

Table 31-37. Commands for MMC/SD/SDIO/CE-ATA Cards (continued)

CMD INDEX	Type	Argument	Response	Abbreviation	Description
ACMD23[4]	ac	—	R1	SET_WR_BLK_ERASE_COUNT	—
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating conditions register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	—	R1	SET_CLR_CARD_DETECT	—
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1 CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).

2 CMD6 differs completely between high-speed MMC cards and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.

3 Command SWITCH is for high-speed MMC cards as well as for CE-ATA cards over the MMC interface. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 31-38.

4 ACMDs is preceded with the APP_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).

The access bits for the EXT_CSD Access Modes are shown in Table 31-38.

Table 31-38. EXT_CSD Access Modes

Bits	Access Name	Operation
00	Command Set	The command set is changed according to the Cmd Set field of the argument
01	Set Bits	The bits in the pointed byte are set, according to the 1 bits in the Value field.
10	Clear Bits	The bits in the pointed byte are cleared, according to the 1 bits in the Value field.
11	Write Byte	The Value field is written into the pointed byte.

31.7 Software Restrictions

This section describes the software restrictions.

31.7.1 Initialization Active

The driver cannot set the INITA bit in the system control register when either of the command line or data lines is active, so the driver is responsible to ensure both CDIHB and CIHB bits are cleared. In order to auto clear the INITA bit, the SDCLKEN bit must be 1, otherwise no clock signal goes out to the card and INITA never clears.

31.7.1.1 Software Polling Procedure

For polling read or write, once the software begins a buffer read or write, it must access exactly the number of times as the values set in the watermark level register; moreover, if the block size is not the times of the value in watermark level register (read and write respectively), the software must access exactly the remained number of words at the end of each block. For example, for read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

31.7.1.2 Suspend Operation

In order to suspend the data transfer, the software must inform eSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform eSDHC that the transfer is suspended.

If the software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remained number of blocks before sending the normal command marked as suspend, otherwise on sending such 'suspend' command, eSDHC regards that the current transfer is aborted and changes BLKCNT register to its original value, instead of keeping the remained number of blocks.

31.7.1.3 Data Length Setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be times of 4.

31.7.1.4 (A)DMA Address Setting

To configure ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so software must make sure TC bit is cleared prior to configuring ADMA1/ADMA2/DMA address register.

31.7.1.5 Data Port Access

Data Port does not support parallel access. For example, during an external DMA access, it is not allowed to write any data to the Data Port by CPU; or during a CPU read operation, it is also prohibited to write any data to the Data Port, by either CPU or external DMA. Otherwise the data would be corrupted inside the eSDHC buffer.

31.7.1.6 Change Clock Frequency

eSDHC does not automatically gates off the card clock when the host driver changes the clock frequency. To remove possible glitch on the card clock, clear SDCLKEN bit when changing clock divisor value and set SDCLKEN bit to '1' after SDSTB bit is '1' again.

Chapter 32
Fast Ethernet Controller (FEC)

32.1 Introduction

This chapter provides a feature-set overview, a functional block diagram, and transceiver connection information for both the 10 and 100 Mbps Media Independent Interface (MII), as well as the 7-wire serial interface. Additionally, detailed descriptions of operation and the programming model are included.

32.2 Overview

The Fast Ethernet Controller (FEC) is designed to support both 10 and 100 Mbps Ethernet/IEEE 802.3 networks. An external transceiver interface and transceiver function are required to complete the interface to the media. The FEC supports three different standard physical interfaces (MAC-PHY) for connection to an external Ethernet transceiver.

32.2.1 Features

The FEC incorporates the following features:

- Support for three different Ethernet physical interfaces:
  - 100-Mbps IEEE 802.3 MII
  - 10-Mbps IEEE 802.3 MII
  - 10-Mbps 7-wire interface (industry standard)
- IEEE 802.3 full duplex flow control
- Programmable max frame length supports IEEE 802.1 VLAN tags and priority
- Support for full-duplex operation (200 Mbps throughput) with a minimum system clock rate of 50 MHz
- Support for half-duplex operation (100 Mbps throughput) with a minimum system clock rate of 25 MHz
- Retransmission from transmit FIFO following a collision (no processor bus utilization)
- Automatic internal flushing of the receive FIFO for runts (collision fragments) and address recognition rejects (no processor bus utilization)
- Address recognition
  - Frames with broadcast address may be always accepted or always rejected
  - Exact match for single 48-bit individual (unicast) address
  - Hash (64-bit hash) check of individual (unicast) addresses

— Hash (64-bit hash) check of group (multicast) addresses
— Promiscuous mode

32.3 Modes of Operation

The primary operational modes are described in this section.

32.3.1 Full- and Half-Duplex Operation

Full-duplex mode is intended for use on point to point links between switches or end node to switch. Half-duplex mode is used in connections between an end node and a repeater or between repeaters. Selection of the duplex mode is controlled by the TCR[FDEN] bit. When configured for full-duplex mode, flow control may be enabled, which is affected by the TCR[RFC_PAUSE], TCR[TFC_PAUSE], and RCR[FCE] bits. See Section 32.5.7, Full-Duplex Flow Control," for more details.

32.3.2 Interface Options

The following interface options are supported. A detailed discussion of the interface configurations is provided in Section 32.5.2, Network Interface Options."

32.3.2.1 10 Mbps and 100 Mbps MII Interface

MII is the Media Independent Interface defined by the IEEE 802.3 standard for 10/100 Mbps operation. The MAC-PHY interface may be configured to operate in MII mode by asserting RCR[MII_MODE].

The speed of operation is determined by the FEC_TX_CLK and FEC_RX_CLK pins which are driven by the external transceiver. The transceiver will either autonegotiate the speed or control by software via the serial management interface (FEC_MDC/FEC_MDIO) pins to the transceiver. See the descriptions in Section 32.6.4.6, MII Management Frame Register (MMFR)" and Section 32.6.4.7, MII Speed Control Register (MSCR)" respectively, as well as the section on the MII for a description of how to read and write registers in the transceiver via this interface.

32.3.2.2 10 Mbps 7-Wire Interface Operation

The FEC supports a 7-wire interface as used by many 10 Mbps ethernet transceivers. The RCR[MII_MODE] bit controls this functionality. If this bit is deasserted, the MII mode is disabled and the 10 Mbps, 7-wire mode is enabled.

32.3.3 Address Recognition Options

The address options supported are promiscuous, broadcast reject, individual address (hash or exact match), and multicast hash match. Address recognition options are discussed in detail in Section 32.5.5, Ethernet Address Recognition."

32.3.4 Internal Loopback

Internal loopback mode is selected via RCR[LOOP]. Loopback mode is discussed in detail in Section 32.5.10, Internal and External Loopback."

32.4 FEC Top-Level Functional Diagram

The block diagram of the FEC is shown below. The FEC is implemented with a combination of hardware and microcode. The off-chip (Ethernet) interfaces are compliant with industry and IEEE 802.3 standards.

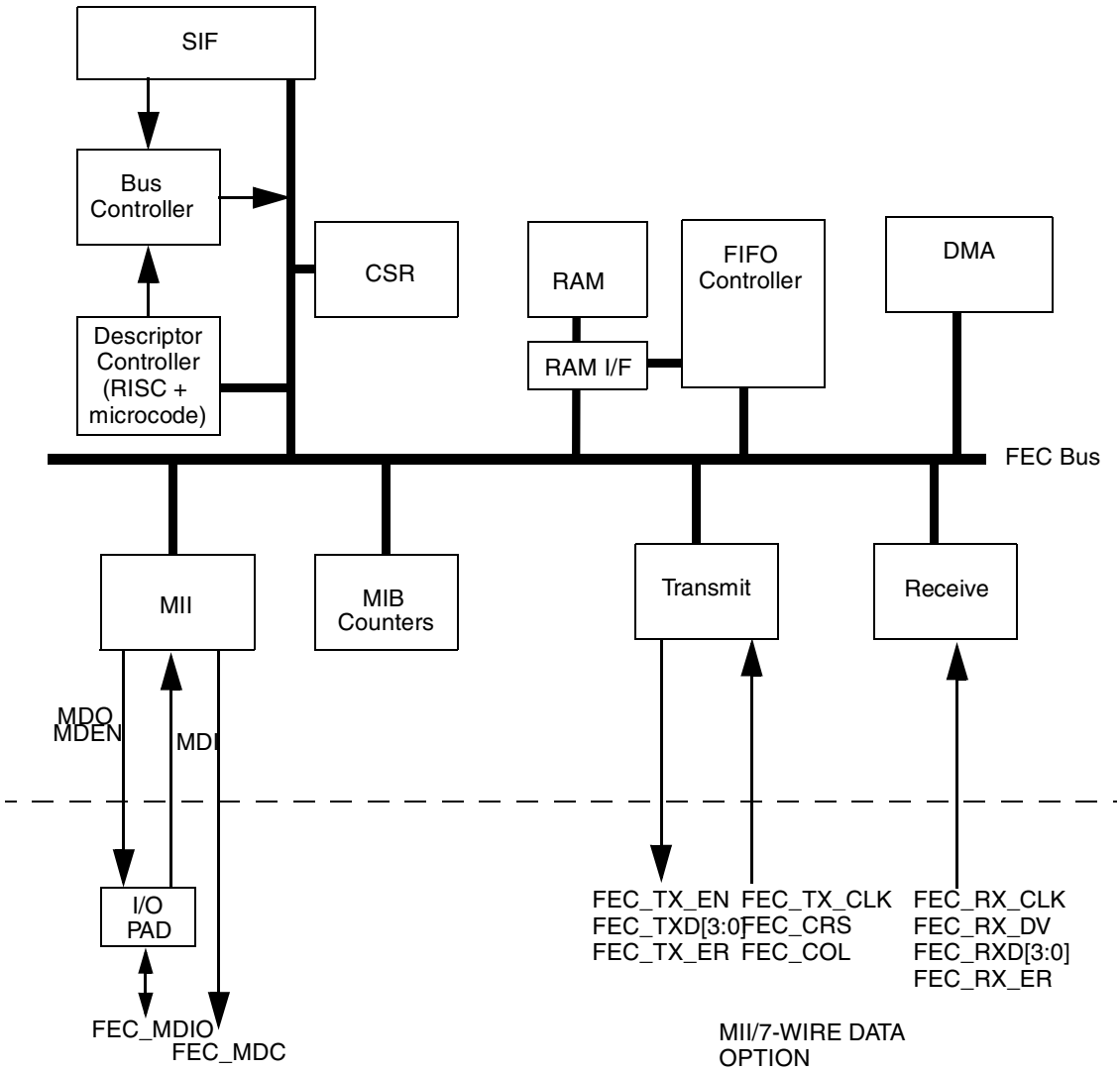


Figure 32-1. FEC Block Diagram

The descriptor controller is a RISC-based controller that provides the following functions in the FEC:

- Initialization (those internal registers not initialized by the user or hardware)
- High level control of the DMA channels (initiating DMA transfers)
- Interpreting buffer descriptors

- Address recognition for receive frames
- Random number generation for transmit collision backoff timer

NOTE

DMA references in this section refer to the FEC's DMA engine. This DMA engine is for the transfer of FEC data only, and is not related to the system DMA controller.

The RAM is the focal point of all data flow in the Fast Ethernet Controller and is divided into transmit and receive FIFOs. The FIFO boundaries are programmable using the FRSR register. User data flows to/from the DMA block from/to the receive/transmit FIFOs. Transmit data flows from the transmit FIFO into the transmit block and receive data flows from the receive block into the receive FIFO.

The user controls the FEC by writing, through the Slave Interface (SIF) module, into control registers located in each block. The Control and Status Register (CSR) block provides global control (for example, Ethernet reset and enable) and interrupt handling registers.

The MII block provides a serial channel for control/status communication with the external physical layer device (transceiver). This serial channel consists of the FEC_MDC (Management Data Clock) and FEC_MDIO (Management Data Input/Output) lines of the MII interface.

The DMA block provides multiple channels allowing transmit data, transmit descriptor, receive data and receive descriptor accesses to run independently.

The Transmit and Receive blocks provide the Ethernet MAC functionality (with some assist from microcode).

The Message Information Block (MIB) maintains counters for a variety of network events and statistics. It is not necessary for operation of the FEC but provides valuable counters for network management. The counters supported are the RMON (RFC 1757) Ethernet Statistics group and some of the IEEE 802.3 counters. See Section 32.6.3, MIB Block Counters Memory Map," for more information.

32.5 Functional Description

This section describes the operation of the FEC, beginning with the hardware and software initialization sequence, followed by a detailed description of the functions of the FEC.

32.5.1 Initialization Sequence

This section describes which registers are reset due to hardware reset, which are reset by the FEC RISC, and which locations the user must initialize prior to enabling the FEC.

32.5.1.1 Hardware Controlled Initialization

In the FEC, registers and control logic that generate interrupts are reset by hardware. A hardware reset deasserts output signals and resets general configuration bits.

Other registers are reset when the ECR[ETHER_EN] bit is cleared, as indicated in Table 32-1. ECR[ETHER_EN] is deasserted by a hard reset or may be deasserted by software to halt operation. By

deasserting ECR[ETHER_EN], the configuration control registers such as the TCR and RCR will not be reset, but the entire data path will be reset.

Table 32-1. ECR[ETHER_EN] De-Assertion Effect on FEC

Register/Machine	Reset Value
XMIT block	Transmission is aborted (bad CRC appended)
RECV block	Receive activity is aborted
DMA block	All DMA activity is terminated
RDAR	Cleared
TDAR	Cleared
Descriptor Controller block	Halt operation

32.5.1.2 User Initialization (Prior to Asserting ECR[ETHER_EN])

The user needs to initialize portions the FEC prior to setting the ECR[ETHER_EN] bit. The exact values will depend on the particular application. The order of initializations is not important.

FEC registers requiring initialization are defined in Table 32-2.

Table 32-2. User Initialization (Before ECR[ETHER_EN])

Description
Initialize EIMR
Clear EIR (write 0xFFFF_FFFF)
TFWR (optional)
IALR / IAUR
GAUR / GALR
PALR / PAUR
OPD (only needed for full duplex flow control)
RCR
TCR
MSCR (optional)
Clear MIB_RAM (locations 0xC003_8000 + 0x200-0x2FC)

FEC FIFO/DMA registers that require initialization are defined in Table 32-3.

Table 32-3. FEC User Initialization (Before ECR[ETHER_EN])

Description
Initialize FRSR (optional)
Initialize EMRBR
Initialize ERDSR
Initialize ETDSR
Initialize (Empty) Transmit Descriptor ring
Initialize (Empty) Receive Descriptor ring

32.5.1.3 Microcontroller Initialization

In the FEC, the descriptor control RISC initializes some registers after ECR[ETHER_EN] is asserted. After the microcontroller initialization sequence is complete, the hardware is ready for operation.

Table 32-4 shows microcontroller initialization operations.

Table 32-4. Microcontroller Initialization

Description
Initialize BackOff Random Number Seed
Activate Receiver
Activate Transmitter
Clear Transmit FIFO
Clear Receive FIFO
Initialize Transmit Ring Pointer
Initialize Receive Ring Pointer
Initialize FIFO Count Registers

32.5.1.4 User Initialization (After Asserting ECR[ETHER_EN])

After asserting ECR[ETHER_EN], the user can set up the buffer/frame descriptors and write to the TDAR and RDAR. See Section 32.6.5, Buffer Descriptors," for more details.

32.5.2 Network Interface Options

The FEC supports both an MII interface for 10/100 Mbps Ethernet and a 7-wire serial interface for 10 Mbps Ethernet. The interface mode is selected by the RCR[MII_MODE] bit. In MII mode (RCR[MII_MODE] = 1), there are 18 signals defined by the IEEE 802.3 standard and supported by the FEC. These signals are shown in Table 32-5 below.

Table 32-5. MII Mode

Signal Description	FEC pin	Direction
Transmit Clock	FEC_TX_CLK	In
Transmit Enable	FEC_TX_EN	Out
Transmit Data	FEC_TXD[3:0]	Out
Transmit Error	FEC_TX_ER	Out
Collision	FEC_COL	In
Carrier Sense	FEC_CRS	In
Receive Clock	FEC_RX_CLK	In
Receive Data Valid	FEC_RX_DV	In
Receive Data	FEC_RXD[3:0]	In

Table 32-5. MII Mode (continued)

Signal Description	FEC pin	Direction
Receive Error	FEC_RX_ER	In
Management Data Clock	FEC_MDC	Out
Management Data Input/Output	FEC_MDIO	I/O

The 7-wire serial mode interface (RCR[MII_MODE] = 0) operates in what is generally referred to as the "AMD" mode. 7-wire mode connections to the external transceiver are shown in Table 32-6.

Table 32-6. 7-Wire Mode Configuration

Signal description	FEC Pin	Direction
Transmit Clock	FEC_TX_CLK	In
Transmit Enable	FEC_TX_EN	Out
Transmit Data	FEC_TXD[0]	Out
Collision	FEC_COL	In
Receive Clock	FEC_RX_CLK	In
Receive Data Valid	FEC_RX_DV	In
Receive Data	FEC_RXD[0]	In

32.5.3 FEC Frame Transmission

The Ethernet transmitter is designed to work with almost no intervention from software. Once ECR[ETHER_EN] is asserted and data appears in the transmit FIFO, the FEC is able to transmit onto the network.

When the transmit FIFO fills to the watermark (defined by the TFWR register), the FEC transmit logic will assert FEC_TX_EN and start transmitting the preamble (PA) sequence, the start frame delimiter (SFD), and then the frame information from the FIFO. However, the controller defers the transmission if the network is busy (FEC_CRS asserts). Before transmitting, the controller waits for carrier sense to become inactive, then determines if carrier sense stays inactive for 60 bit times. If so, the transmission begins after waiting an additional 36 bit times (96 bit times after carrier sense originally became inactive). See Section 32.5.11.1, Transmission Errors" for more details.

If a collision occurs during transmission of the frame (half duplex mode), the Ethernet controller follows the specified backoff procedures and attempts to retransmit the frame until the retry limit is reached. The transmit FIFO stores at least the first 64 bytes of the transmit frame, so that they do not have to be retrieved from system memory in case of a collision. This improves bus utilization and latency in case immediate retransmission is necessary.

When all the frame data has been transmitted, the FCS (Frame Check Sequence or 32-bit Cyclic Redundancy Check, CRC) bytes are appended if the TC bit is set in the transmit frame control word. If the ABC bit is set in the transmit frame control word, a bad CRC will be appended to the frame data regardless of the TC bit value. Following the transmission of the CRC, the Ethernet controller writes the frame status

information to the MIB block. Short frames are automatically padded by the transmit logic (if the TC bit in the transmit buffer descriptor for the end of frame buffer = 1).

Both buffer (TXB) and frame (TXF) interrupts may be generated as determined by the settings in the EIMR.

The transmit error interrupts are HBERR, BABT, LATE_COL, COL_RETRY_LIM, and XFIFO_UN. If the transmit frame length exceeds MAX_FL bytes the BABT interrupt will be asserted, however the entire frame will be transmitted (no truncation).

To pause transmission, set the GTS (graceful transmit stop) bit in the TCR register. When the TCR[GTS] is set, the FEC transmitter stops immediately if transmission is not in progress; otherwise, it continues transmission until the current frame either finishes or terminates with a collision. After the transmitter has stopped the GRA (graceful stop complete) interrupt is asserted. If TCR[GTS] is cleared, the FEC resumes transmission with the next frame.

The Ethernet controller transmits bytes least significant bit first.

32.5.4 FEC Frame Reception

The FEC receiver is designed to work with almost no intervention from the host and can perform address recognition, CRC checking, short frame checking, and maximum frame length checking.

When the driver enables the FEC receiver by asserting ECR[ETHER_EN], it will immediately start processing receive frames. When FEC_RX_DV asserts, the receiver will first check for a valid PA/SFD header. If the PA/SFD is valid, it will be stripped and the frame will be processed by the receiver. If a valid PA/SFD is not found, the frame will be ignored.

In serial mode, the first 16 bit times of RX_D0 following assertion of FEC_RX_DV are ignored. Following the first 16 bit times the data sequence is checked for alternating 1/0s. If a 11 or 00 data sequence is detected during bit times 17 to 21, the remainder of the frame is ignored. After bit time 21, the data sequence is monitored for a valid SFD (11). If a 00 is detected, the frame is rejected. When a 11 is detected, the PA/SFD sequence is complete.

In MII mode, the receiver checks for at least one byte matching the SFD. Zero or more PA bytes may occur, but if a 00 bit sequence is detected prior to the SFD byte, the frame is ignored.

After the first 6 bytes of the frame have been received, the FEC performs address recognition on the frame.

Once a collision window (64 bytes) of data has been received and if address recognition has not rejected the frame, the receive FIFO is signalled that the frame is “accepted” and may be passed on to the DMA. If the frame is a runt (due to collision) or is rejected by address recognition, the receive FIFO is notified to “reject” the frame. Thus, no collision fragments are presented to the user except late collisions, which indicate serious LAN problems.

During reception, the Ethernet controller checks for various error conditions and once the entire frame is written into the FIFO, a 32-bit frame status word is written into the FIFO. This status word contains the M, BC, MC, LG, NO, CR, OV and TR status bits, and the frame length. See Section 32.5.11.2, Reception Errors,” for more details.

Receive Buffer (RXB) and Frame Interrupts (RXF) may be generated if enabled by the EIMR register. A receive error interrupt is babbling receiver error (BABR). Receive frames are not truncated if they exceed the max frame length (MAX_FL); however, the BABR interrupt will occur and the LG bit in the Receive Buffer Descriptor (RxBD) will be set. See Section 32.6.5.2, Ethernet Receive Buffer Descriptor (RxBD)," for more details.

When the receive frame is complete, the FEC sets the L-bit in the RxBD, writes the other frame status bits into the RxBD, and clears the E-bit. The Ethernet controller next generates a maskable interrupt (RXF bit in EIR, maskable by RXF bit in EIMR), indicating that a frame has been received and is in memory. The Ethernet controller then waits for a new frame.

The Ethernet controller receives serial data LSB first.

32.5.5 Ethernet Address Recognition

The FEC filters the received frames based on Destination Address (DA) type — individual (unicast), group (multicast), or broadcast (all-ones group address). The difference between an individual address and a group address is determined by the I/G bit in the destination address field. A flowchart for address recognition on received frames is illustrated in the figures below.

Address recognition is accomplished through the use of the receive block and microcode running on the microcontroller. The flowchart shown in Figure 32-2 illustrates the address recognition decisions made by the receive block, while Figure 32-3 illustrates the decisions made by the microcontroller.

If the DA is a broadcast address and broadcast reject (RCR[BC_REJ]) is deasserted, then the frame will be accepted unconditionally, as shown in Figure 32-2. Otherwise, if the DA is not a broadcast address, then the microcontroller runs the address recognition subroutine, as shown in Figure 32-3.

If the DA is a group (multicast) address and flow control is disabled, then the microcontroller will perform a group hash table lookup using the 64-entry hash table programmed in GAUR and GALR. If a hash match occurs, the receiver accepts the frame.

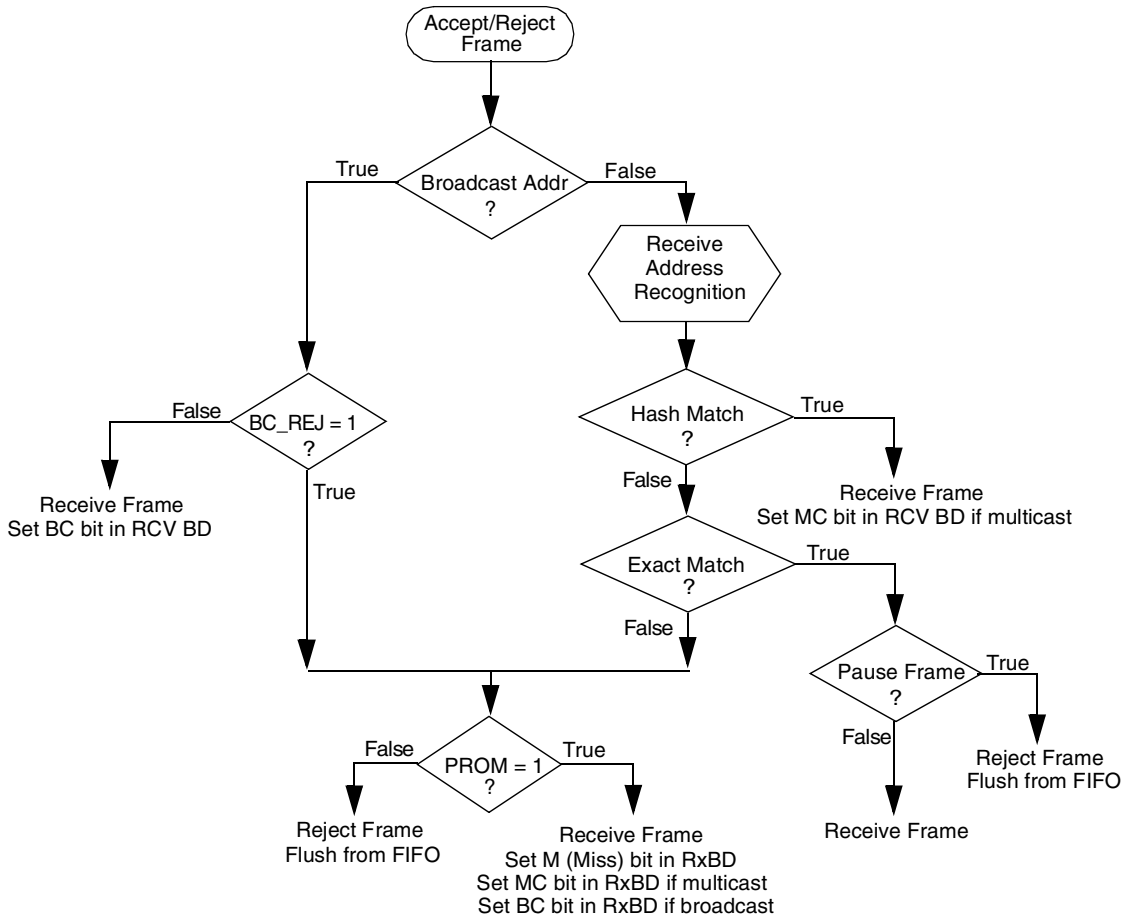
If flow control is enabled, the microcontroller will do an exact address match check between the DA and the designated PAUSE DA (01:80:C2:00:00:01). If the receive block determines that the received frame is a valid PAUSE frame, then the frame will be rejected. Note the receiver will detect a PAUSE frame with the DA field set to either the designated PAUSE DA or the unicast physical address.

If the DA is the individual (unicast) address, the microcontroller performs an individual exact match comparison between the DA and 48-bit physical address that the user programs in the PALR and PAUR registers. If an exact match occurs, the frame is accepted; otherwise, the microcontroller does an individual hash table lookup using the 64-entry hash table programmed in registers, IAUR and IALR. In the case of an individual hash match, the frame is accepted. Again, the receiver will accept or reject the frame based on PAUSE frame detection, shown in Figure 32-2.

If neither a hash match (group or individual), nor an exact match occur, then if promiscuous mode is enabled (RCR[PROM] = 1), then the frame will be accepted and the MISS bit in the receive buffer descriptor is set; otherwise, the frame will be rejected.

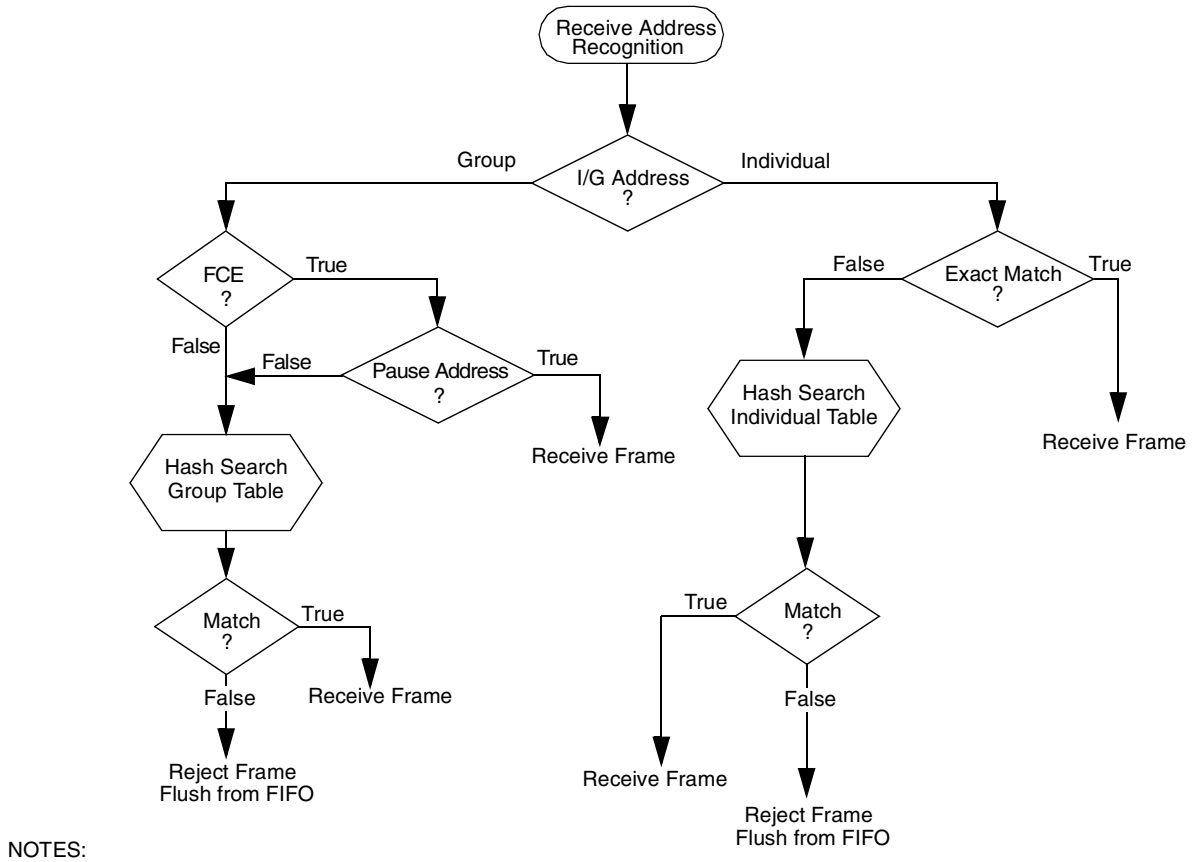
Similarly, if the DA is a broadcast address, broadcast reject (RCR[BC_REJ]) is asserted, and promiscuous mode is enabled, then the frame will be accepted and the MISS bit in the receive buffer descriptor is set; otherwise, the frame will be rejected.

In general, when a frame is rejected, it is flushed from the FIFO.



NOTES:
BC_REJ—Field in RCR register (BroadCast REJect)
PROM—Field in RCR register (PROMiscuous mode)
Pause Frame—Valid PAUSE frame received

Figure 32-2. Ethernet Address Recognition—Receive Block Decisions



NOTES:
FCE - field in RCR register (Flow Control Enable)
I/G - Individual/Group bit in Destination Address (least significant bit in first byte received in MAC frame)

Figure 32-3. Ethernet Address Recognition—Microcode Decisions

32.5.6 Hash Algorithm

The hash table algorithm used in the group and individual hash filtering operates as follows. The 48-bit destination address is mapped into one of 64 bits, which are represented by 64 bits stored in GAUR, GALR (group address hash match) or IAUR, IALR (individual address hash match). This mapping is performed by passing the 48-bit address through the on-chip 32-bit CRC generator and selecting the 6 most significant bits of the CRC-encoded result to generate a number between 0 and 63. The MSB of the CRC result selects GAUR (MSB = 1) or GALR (MSB = 0). The least significant 5 bits of the hash result select the bit within the selected register. If the CRC generator selects a bit that is set in the hash table, the frame is accepted; otherwise, it is rejected.For example, if eight group addresses are stored in the hash table and random group addresses are received, the hash table prevents roughly 56/64 (or 87.5%) of the group address frames from reaching memory. Those that do reach memory must be further filtered by the processor to determine if they truly contain one of the eight desired addresses.

The effectiveness of the hash table declines as the number of addresses increases.

The hash table registers must be initialized by the user. The CRC32 polynomial to use in computing the hash is:

$$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

A table of example Destination Addresses and corresponding hash values is included below for reference.

Table 32-7. Destination Address to 6-Bit Hash

48-bit DA	6-bit Hash (in hex)	Hash Decimal Value
65:FF:FF:FF:FF:FF	0x0	0
55:FF:FF:FF:FF:FF	0x1	1
15:FF:FF:FF:FF:FF	0x2	2
35:FF:FF:FF:FF:FF	0x3	3
B5:FF:FF:FF:FF:FF	0x4	4
95:FF:FF:FF:FF:FF	0x5	5
D5:FF:FF:FF:FF:FF	0x6	6
F5:FF:FF:FF:FF:FF	0x7	7
DB:FF:FF:FF:FF:FF	0x8	8
FB:FF:FF:FF:FF:FF	0x9	9
BB:FF:FF:FF:FF:FF	0xA	10
8B:FF:FF:FF:FF:FF	0xB	11
0B:FF:FF:FF:FF:FF	0xC	12
3B:FF:FF:FF:FF:FF	0xD	13
7B:FF:FF:FF:FF:FF	0xE	14
5B:FF:FF:FF:FF:FF	0xF	15
27:FF:FF:FF:FF:FF	0x10	16
07:FF:FF:FF:FF:FF	0x11	17
57:FF:FF:FF:FF:FF	0x12	18
77:FF:FF:FF:FF:FF	0x13	19
F7:FF:FF:FF:FF:FF	0x14	20
C7:FF:FF:FF:FF:FF	0x15	21
97:FF:FF:FF:FF:FF	0x16	22
A7:FF:FF:FF:FF:FF	0x17	23
99:FF:FF:FF:FF:FF	0x18	24
B9:FF:FF:FF:FF:FF	0x19	25
F9:FF:FF:FF:FF:FF	0x1A	26

Table 32-7. Destination Address to 6-Bit Hash (continued)

48-bit DA	6-bit Hash (in hex)	Hash Decimal Value
C9:FF:FF:FF:FF:FF	0x1B	27
59:FF:FF:FF:FF:FF	0x1C	28
79:FF:FF:FF:FF:FF	0x1D	29
29:FF:FF:FF:FF:FF	0x1E	30
19:FF:FF:FF:FF:FF	0x1F	31
D1:FF:FF:FF:FF:FF	0x20	32
F1:FF:FF:FF:FF:FF	0x21	33
B1:FF:FF:FF:FF:FF	0x22	34
91:FF:FF:FF:FF:FF	0x23	35
11:FF:FF:FF:FF:FF	0x24	36
31:FF:FF:FF:FF:FF	0x25	37
71:FF:FF:FF:FF:FF	0x26	38
51:FF:FF:FF:FF:FF	0x27	39
7F:FF:FF:FF:FF:FF	0x28	40
4F:FF:FF:FF:FF:FF	0x29	41
1F:FF:FF:FF:FF:FF	0x2A	42
3F:FF:FF:FF:FF:FF	0x2B	43
BF:FF:FF:FF:FF:FF	0x2C	44
9F:FF:FF:FF:FF:FF	0x2D	45
DF:FF:FF:FF:FF:FF	0x2E	46
EF:FF:FF:FF:FF:FF	0x2F	47
93:FF:FF:FF:FF:FF	0x30	48
B3:FF:FF:FF:FF:FF	0x31	49
F3:FF:FF:FF:FF:FF	0x32	50
D3:FF:FF:FF:FF:FF	0x33	51
53:FF:FF:FF:FF:FF	0x34	52
73:FF:FF:FF:FF:FF	0x35	53
23:FF:FF:FF:FF:FF	0x36	54
13:FF:FF:FF:FF:FF	0x37	55
3D:FF:FF:FF:FF:FF	0x38	56
0D:FF:FF:FF:FF:FF	0x39	57
5D:FF:FF:FF:FF:FF	0x3A	58
7D:FF:FF:FF:FF:FF	0x3B	59

Table 32-7. Destination Address to 6-Bit Hash (continued)

48-bit DA	6-bit Hash (in hex)	Hash Decimal Value
FD:FF:FF:FF:FF:FF	0x3C	60
DD:FF:FF:FF:FF:FF	0x3D	61
9D:FF:FF:FF:FF:FF	0x3E	62
BD:FF:FF:FF:FF:FF	0x3F	63

32.5.7 Full-Duplex Flow Control

Full-duplex flow control allows the user to transmit pause frames and to detect received pause frames. Upon detection of a pause frame, FEC data frame transmission stops for a given pause duration.

To enable pause frame detection, the FEC must operate in full-duplex mode (TCR[FDEN] asserted) and flow control enable (RCR[FCE]) must be asserted. The FEC detects a pause frame when the fields of the incoming frame match the pause frame specifications, as shown Table 32-8. In addition, the receive status associated with the frame should indicate that the frame is valid.

Table 32-8. PAUSE Frame Field Specification

48-bit Destination Address	0x0180_c200_0001 or Physical Address
48-bit Source Address	Any
16-bit Type	0x8808
16-bit Opcode	0x0001
16-bit PAUSE Duration	0x0000 to 0xFFFF

Pause frame detection is performed by the receiver and microcontroller modules. The microcontroller runs an address recognition subroutine to detect the specified pause frame destination address, while the receiver detects the type and opcode pause frame fields. On detection of a pause frame, TCR[GTS] is asserted by the FEC internally. When transmission has paused, the EIR[GRA] interrupt is asserted and the pause timer begins to increment. Note that the pause timer makes use of the transmit backoff timer hardware, which is used for tracking the appropriate collision backoff time in half-duplex mode. The pause timer increments once every slot time, until OPD[PAUSE_DUR] slot times have expired. On OPD[PAUSE_DUR] expiration, TCR[GTS] is deasserted allowing FEC data frame transmission to resume. Note that the receive flow control pause (TCR[RFC_PAUSE]) status bit is asserted while the transmitter is paused due to reception of a pause frame.

To transmit a pause frame, the FEC must operate in full-duplex mode and the user must assert flow control pause (TCR[TFC_PAUSE]). On assertion of transmit flow control pause (TCR[TFC_PAUSE]), the transmitter asserts TCR[GTS] internally. When the transmission of data frames stops, the EIR[GRA] (graceful stop complete) interrupt asserts. Following EIR[GRA] assertion, the pause frame is transmitted. On completion of pause frame transmission, flow control pause (TCR[TFC_PAUSE]) and TCR[GTS] are deasserted internally.

The user must specify the desired pause duration in the OPD register.

Note that when the transmitter is paused due to receiver/microcontroller pause frame detection, transmit flow control pause (TCR[TFC_PAUSE]) still may be asserted, which causes the transmission of a single pause frame. In this case, the EIR[GRA] interrupt is not asserted.

32.5.8 Interpacket Gap (IPG) Time

The minimum interpacket gap (IPG) time for back-to-back transmission is 96 bit times. After completing a transmission or after the backoff algorithm completes, the transmitter waits for carrier sense to be negated before starting its 96 bit time IPG counter. Frame transmission may begin 96 bit times after carrier sense is negated if it stays negated for at least 60 bit times. If carrier sense asserts during the last 36 bit times, it will be ignored and a collision will occur.

The receiver receives back-to-back frames with a minimum spacing of at least 28 bit times. If an interpacket gap between receive frames is less than 28 bit times, the following frame may be discarded by the receiver.

32.5.9 Collision Handling

If a collision occurs during frame transmission, the Ethernet controller will continue the transmission for at least 32 bit times, transmitting a JAM pattern consisting of 32 ones. If the collision occurs during the preamble sequence, the JAM pattern will be sent after the end of the preamble sequence.

If a collision occurs within 512 bit times, the retry process is initiated. The transmitter waits a random number of slot times, where one slot time is 512 bit times. If a collision occurs after 512 bit times, then no retransmission is performed and the end of frame buffer is closed with a Late Collision (LC) error indication.

32.5.10 Internal and External Loopback

Both internal and external loopback are supported by the Ethernet controller. In loopback mode, both of the FIFOs are used and the FEC actually operates in a full-duplex fashion. Both internal and external loopback are configured using combinations of the LOOP and DRT bits in the RCR register and the FDEN bit in the TCR register.

For both internal and external loopback set FDEN = 1.

For internal loopback set RCR[LOOP] = 1 and RCR[DRT] = 0. FEC_TX_EN and FEC_TX_ER will not assert during internal loopback. During internal loopback, the transmit/receive data rate is higher than in normal operation because the internal system clock is used by the transmit and receive blocks instead of the clocks from the external transceiver. This causes an increase in the required system bus bandwidth for transmit and receive data being transferred to and from external memory via DMA. It may be necessary to pace the frames on the transmit side and/or limit the size of the frames to prevent transmit FIFO underrun and receive FIFO overflow.

For external loopback set RCR[LOOP] = 0, RCR[DRT] = 0 and configure the external transceiver for loopback.

32.5.11 Ethernet Error-Handling Procedure

The Ethernet controller reports frame reception and transmission error conditions using the FEC RxBDs, the EIR register, and the MIB block counters.

32.5.11.1 Transmission Errors

There are four types of transmission errors:

- Transmitter underrun
- Retransmission attempts limit expired
- Late collision
- Heartbeat

32.5.11.1.1 Transmitter Underrun

If this error occurs, the FEC sends 32 bits that ensure a CRC error and stops transmitting. All remaining buffers for that frame are then flushed and closed. The UN bit is set in the EIR. The FEC will then continue to the next transmit buffer descriptor and begin transmitting the next frame.

The “UN” interrupt will be asserted if enabled in the EIMR register.

32.5.11.1.2 Retransmission Attempts Limit Expired

When this error occurs, the FEC terminates transmission. All remaining buffers for that frame are flushed and closed, and the RL bit is set in the EIR. The FEC will then continue to the next transmit buffer descriptor and begin transmitting the next frame.

The “RL” interrupt will be asserted if enabled in the EIMR register.

32.5.11.1.3 Late Collision

When a collision occurs after the slot time (512 bits starting at the Preamble), the FEC terminates transmission. All remaining buffers for that frame are flushed and closed, and the LC bit is set in the EIR register. The FEC will then continue to the next transmit buffer descriptor and begin transmitting the next frame.

The LC interrupt is asserted if enabled in the EIMR register.

32.5.11.1.4 Heartbeat

Some transceivers have a self-test feature called “heartbeat” or “signal quality error.” To signify a good self-test, the transceiver indicates a collision to the FEC within 4 microseconds after completion of a frame transmitted by the Ethernet controller. This indication of a collision does not imply a real collision error on the network, but is rather an indication that the transceiver still seems to be functioning properly. This is called the heartbeat condition.

If the HBC bit is set in the TCR register and the heartbeat condition is not detected by the FEC after a frame transmission, then a heartbeat error occurs. When this error occurs, the FEC closes the buffer, sets the HB bit in the EIR register, and generates the HBERR interrupt if it is enabled.

32.5.11.2 Reception Errors

There are five types of reception errors:

- Overrun
- Non-octet (dribbling bits)
- CRC
- Frame-length violation
- Truncation

32.5.11.2.1 Overrun

If the receive block has data to put into the receive FIFO and the receive FIFO is full, the FEC sets the OV bit in the RxBD. All subsequent data in the frame will be discarded and subsequent frames may also be discarded until the receive FIFO is serviced by the DMA and space is made available. At this point the receive frame/status word is written into the FIFO with the OV bit set. This frame must be discarded by the driver.

32.5.11.2.2 Non-Octet (Dribbling Bits)

The Ethernet controller handles up to seven dribbling bits when the receive frame terminates past an non-octet aligned boundary. Dribbling bits are not used in the CRC calculation. If there is a CRC error, then the frame non-octet aligned (NO) error is reported in the RxBD. If there is no CRC error, then no error is reported.

32.5.11.2.3 CRC

When a CRC error occurs with no dribble bits, the FEC closes the buffer and sets the CR bit in the RxBD. CRC checking cannot be disabled, but the CRC error can be ignored if checking is not required.

32.5.11.2.4 Frame Length Violation

When the receive frame length exceeds MAX_FL bytes the BABR interrupt will be generated, and the LG bit in the end of frame RxBD will be set. The frame is not truncated unless the frame length exceeds 2047 bytes).

32.5.11.2.5 Truncation

When the receive frame length exceeds 2047 bytes, the frame is truncated and the TR bit is set in the RxBD.

32.6 Programming Model

This section provides an overview of the registers, followed by a description of the buffers.

The FEC is programmed by a combination of control and status registers (CSRs) and buffer descriptors. The CSRs are used for mode control and to extract global status information. The descriptors are used to pass data buffers and related buffer information between the hardware and software.

32.6.1 Top Level Module Memory Map

The FEC implementation requires a 1Kbyte memory map space. This is divided into 2 sections of 512 bytes each. The first is used for control/status registers. The second contains event/statistic counters held in the MIB block. Table 32-9 defines the top level memory map.

Table 32-9. Module Memory Map[1]

Address	Function
0xBASE + 0x000-1FF	Control/Status Registers
0xBASE + 0x200-3FF	MIB Block Counters

[1] See Chapter 2, "Memory Map,"for the value of BASE - FEC base address.

32.6.2 Detailed Memory Map (Control/Status Registers)

Table 32-10 shows the FEC register memory map with each register address, name, and a brief description.

Table 32-10. FEC Register Memory Map

Offset	Name	Width	Description
0x004	EIR	32	Interrupt Event Register
0x008	EIMR	32	Interrupt Mask Register
0x010	RDAR	32	Receive Descriptor Active Register
0x014	TDAR	32	Transmit Descriptor Active Register
0x024	ECR	32	Ethernet Control Register
0x040	MMFR	32	MII Management Frame Register
0x044	MSCR	32	MII Speed Control Register
0x064	MIBC	32	MIB Control/Status Register
0x084	RCR	32	Receive Control Register
0x0C4	TCR	32	Transmit Control Register
0x0E4	PALR	32	Physical Address Low Register
0x0E8	PAUR	32	Physical Address High+ Type Field
0x0EC	OPD	32	Opcode + Pause Duration
0x118	IAUR	32	Upper 32 bits of Individual Hash Table
0x11C	IALR	32	Lower 32 Bits of Individual Hash Table
0x120	GAUR	32	Upper 32 bits of Group Hash Table
0x124	GALR	32	Lower 32 bits of Group Hash Table
0x144	TFWR	32	Transmit FIFO Watermark
0x14C	FRBR	32	FIFO Receive Bound Register
0x150	FRSR	32	FIFO Receive FIFO Start Registers
0x180	ERDSR	32	Pointer to Receive Descriptor Ring

Table 32-10. FEC Register Memory Map (continued)

Offset	Name	Width	Description
0x184	ETDSR	32	Pointer to Transmit Descriptor Ring
0x188	EMRBR	32	Maximum Receive Buffer Size

32.6.3 MIB Block Counters Memory Map

Table 32-11 defines the MIB Counters memory map which defines the locations in the MIB RAM space where hardware maintained counters reside. It is the responsibility of software to poll the counters often enough to ensure that rollover is detected. For example, on a 100 Mbps channel an octets counter could roll over every 5.7 minutes.

These counters fall in the 0x200-0x3FF address offset range, and are divided into RMON counters and IEEE counters, as follows

RMON counters are included which cover the Ethernet Statistics counters defined in RFC 1757. In addition to the counters defined in the Ethernet Statistics group, a counter is included to count truncated frames as the FEC only supports frame lengths up to 2047 bytes. The RMON counters are implemented independently for transmit and receive to insure accurate network statistics when operating in full duplex mode.

IEEE counters are included which support the Mandatory and Recommended counter packages defined in section 5 of ANSI/IEEE Std. 802.3 (1998 edition). The IEEE Basic Package objects are supported by the FEC but do not require counters in the MIB block. In addition, some of the recommended package objects which are supported do not require MIB counters. Counters for transmit and receive full duplex flow control frames are included as well.

Table 32-11. MIB Counters Memory Map

Offset	Mnemonic	Description
0x200	RMON_T_DROP	Count of frames not counted correctly
0x204	RMON_T_PACKETS	RMON Tx packet count
0x208	RMON_T_BC_PKT	RMON Tx Broadcast Packets
0x20C	RMON_T_MC_PKT	RMON Tx Multicast Packets
0x210	RMON_T_CRC_ALIGN	RMON Tx Packets w CRC/Align error
0x214	RMON_T_UNDERSIZE	RMON Tx Packets < 64 bytes, good crc
0x218	RMON_T_OVERSIZE	RMON Tx Packets > MAX_FL bytes, good crc
0x21C	RMON_T_FRAG	RMON Tx Packets < 64 bytes, bad crc
0x220	RMON_T_JAB	RMON Tx Packets > MAX_FL bytes, bad crc
0x224	RMON_T_COL	RMON Tx collision count
0x228	RMON_T_P64	RMON Tx 64 byte packets
0x22C	RMON_T_P65TO127	RMON Tx 65 to 127 byte packets

Table 32-11. MIB Counters Memory Map (continued)

Offset	Mnemonic	Description
0x230	RMON_T_P128TO255	RMON Tx 128 to 255 byte packets
0x234	RMON_T_P256TO511	RMON Tx 256 to 511 byte packets
0x238	RMON_T_P512TO1023	RMON Tx 512 to 1023 byte packets
0x23C	RMON_T_P1024TO2047	RMON Tx 1024 to 2047 byte packets
0x240	RMON_T_P_GTE2048	RMON Tx packets w > 2048 bytes
0x244	RMON_T_OCTETS	RMON Tx Octets
0x248	IEEE_T_DROP	Count of frames not counted correctly
0x24C	IEEE_T_FRAME_OK	Frames Transmitted OK
0x250	IEEE_T_1COL	Frames Transmitted with Single Collision
0x254	IEEE_T_MCOL	Frames Transmitted with Multiple Collisions
0x258	IEEE_T_DEF	Frames Transmitted after Deferral Delay
0x25C	IEEE_T_LCOL	Frames Transmitted with Late Collision
0x260	IEEE_T_EXCOL	Frames Transmitted with Excessive Collisions
0x264	IEEE_T_MACERR	Frames Transmitted with Tx FIFO Underrun
0x268	IEEE_T_CSERR	Frames Transmitted with Carrier Sense Error
0x26C	IEEE_T_SQE	Frames Transmitted with SQE Error
0x270	IEEE_T_FDXFC	Flow Control Pause frames transmitted
0x274	IEEE_T_OCTETS_OK	Octet count for Frames Transmitted w/o Error
0x284	RMON_R_PACKETS	RMON Rx packet count
0x288	RMON_R_BC_PKT	RMON Rx Broadcast Packets
0x28C	RMON_R_MC_PKT	RMON Rx Multicast Packets
0x290	RMON_R_CRC_ALIGN	RMON Rx Packets w CRC/Align error
0x294	RMON_R_UNDERSIZE	RMON Rx Packets < 64 bytes, good crc
0x298	RMON_R_OVERSIZE	RMON Rx Packets > MAX_FL bytes, good crc
0x29C	RMON_R_FRAG	RMON Rx Packets < 64 bytes, bad crc
0x2A0	RMON_R_JAB	RMON Rx Packets > MAX_FL bytes, bad crc
0x2A4	RMON_R_RESVD_0	—
0x2A8	RMON_R_P64	RMON Rx 64 byte packets
0x2AC	RMON_R_P65TO127	RMON Rx 65 to 127 byte packets
0x2B0	RMON_R_P128TO255	RMON Rx 128 to 255 byte packets
0x2B4	RMON_R_P256TO511	RMON Rx 256 to 511 byte packets
0x2B8	RMON_R_P512TO1023	RMON Rx 512 to 1023 byte packets
0x2BC	RMON_R_P1024TO2047	RMON Rx 1024 to 2047 byte packets

Table 32-11. MIB Counters Memory Map (continued)

Offset	Mnemonic	Description
0x2C0	RMON_R_P_GTE2048	RMON Rx packets w > 2048 bytes
0x2C4	RMON_R_OCTETS	RMON Rx Octets
0x2C8	IEEE_R_DROP	Count of frames not counted correctly
0x2CC	IEEE_R_FRAME_OK	Frames Received OK
0x2D0	IEEE_R_CRC	Frames Received with CRC Error
0x2D4	IEEE_R_ALIGN	Frames Received with Alignment Error
0x2D8	IEEE_R_MACERR	Receive FIFO Overflow count
0x2DC	IEEE_R_FDXFC	Flow Control Pause frames received
0x2E0	IEEE_R_OCTETS_OK	Octet count for Frames received w/o Error

32.6.4 Register Descriptions

The following sections describe each register in detail.

32.6.4.1 Ethernet Interrupt Event Register (EIR)

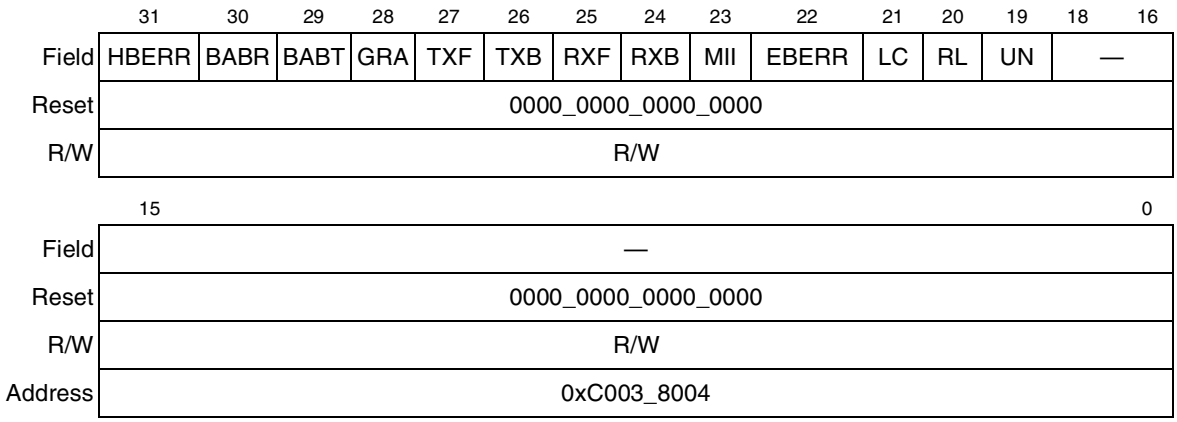
The correspondence between interrupts and EIR bits is shown in Figure 32-4 and Table 32-13 below. When an event occurs that sets a bit in the EIR, an interrupt will be generated if the corresponding bit in the interrupt mask register (EIMR) is also set. The bit in the EIR is cleared if a one is written to that bit position; writing zero has no effect. This register is cleared upon hardware reset.

Interrupts can be divided into operational interrupts, transceiver/network error interrupts, and internal error interrupts. Interrupts which may occur in normal operation are GRA, TXF, TXB, RXF, RXB, and MII. Interrupts resulting from errors/problems detected in the network or transceiver are HBERR, BABR, BABT, LC and RL. Interrupts resulting from internal errors are EBERR and UN.

Some of the error interrupts are independently counted in the MIB block counters. The correspondence between interrupts and counters is shown in Table 32-12. Software may choose to mask off the interrupts since these errors will be visible to network management via the MIB counters.

Table 32-12. Error Interrupts and Block Counters

Interrupt	Counter(s)
HBERR	IEEE_T_SQE
BABR	RMON_R_OVERSIZE (good CRC) RMON_R_JAB (bad CRC)
BABT	RMON_T_OVERSIZE (good CRC), RMON_T_JAB (bad CRC)
LATE_COL	IEEE_T_LCOL
COL_RETRY_LIM	IEEE_T_EXCOL
XFIFO_UN	IEEE_T_MACERR



	31	30	29	28	27	26	25	24	23	22	21	20	19	18	16
Field	HBERR	BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	—	
Reset	0000_0000_0000_0000														
R/W	R/W														

	15	0
Field	—	
Reset	0000_0000_0000_0000	
R/W	R/W	
Address	0xC003_8004	

Figure 32-4. Ethernet Interrupt Event Register (EIR)

Table 32-13. EIR Field Descriptions

Bits	Name	Description
31	HBERR	Heartbeat error. This interrupt indicates that HBC is set in the TCR register and that the COL input was not asserted within the Heartbeat window following a transmission.
30	BABR	Babbling receive error. This bit indicates a frame was received with length in excess of RCR[MAX_FL] bytes.
29	BABT	Babbling transmit error. This bit indicates that the transmitted frame length has exceeded RCR[MAX_FL] bytes. This condition is usually caused by a frame that is too long being placed into the transmit data buffer(s). Truncation does not occur.
28	GRA	Graceful stop complete. This interrupt will be asserted for one of three reasons. Graceful stop means that the transmitter is put into a pause state after completion of the frame currently being transmitted. 1) A graceful stop, which was initiated by the setting of the TCR[GTS] bit is now complete. 2) A graceful stop, which was initiated by the setting of the TCR[TFC_PAUSE] bit is now complete. 3) A graceful stop, which was initiated by the reception of a valid full duplex flow control “pause” frame is now complete. See the “Full Duplex Flow Control” section of the Functional Description chapter.
27	TXF	Transmit frame interrupt. This bit indicates that a frame has been transmitted and that the last corresponding buffer descriptor has been updated.
26	TXB	Transmit buffer interrupt. This bit indicates that a transmit buffer descriptor has been updated.
25	RXF	Receive frame interrupt. This bit indicates that a frame has been received and that the last corresponding buffer descriptor has been updated.
24	RXB	Receive buffer interrupt. This bit indicates that a receive buffer descriptor has been updated that was not the last in the frame.
23	MII	MII interrupt. This bit indicates that the MII has completed the data transfer requested.
22	EBERR	Ethernet bus error. This bit indicates that a system bus error occurred when a DMA transaction was underway. When the EBERR bit is set, ECR[ETHER_EN] will be cleared, halting frame processing by the FEC. When this occurs software will need to insure that the FIFO controller and DMA are also soft reset.

Table 32-13. EIR Field Descriptions (continued)

Bits	Name	Description
21	LC	Late collison. This bit indicates that a collision occurred beyond the collision window (slot time) in half duplex mode. The frame is truncated with a bad CRC and the remainder of the frame is discarded.
20	RL	Collision retry limit. This bit indicates that a collision occurred on each of 16 successive attempts to transmit the frame. The frame is discarded without being transmitted and transmission of the next frame will commence. Can only occur in half duplex mode.
19	UN	Transmit FIFO underrun. This bit indicates that the transmit FIFO became empty before the complete frame was transmitted. A bad CRC is appended to the frame fragment and the remainder of the frame is discarded.
18–0	—	Reserved, should be cleared.

32.6.4.2 Interrupt Mask Register (EIMR)

The EIMR controls which interrupt events are allowed to generate actual interrupts. All implemented bits in this CSR are read/write. This register is cleared upon a hardware reset. If the corresponding bits in both the EIR and EIMR are set, the interrupt will be signalled to the CPU. The interrupt signal will remain asserted until a 1 is written to the EIR bit (write 1 to clear) or a 0 is written to the EIMR bit.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	16
Field	HBERR	BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	—	
Reset	0000_0000_0000_0000														
R/W	R/W														

	15	0
Field	—	
Reset	0000_0000_0000_0000	
R/W	R/W	
Address	0xC003_8008	

Figure 32-5. Interrupt Mask Register (EIMR)

Table 32-14. EIMR Field Descriptions

Bits	Name	Description
31–19	See Figure 32-5 and Table 32-13.	Interrupt Mask. Each bit corresponds to an interrupt source defined by the EIR register. The corresponding EIMR bit determines whether an interrupt condition can generate an interrupt. At every system clock, the EIR samples the signal generated by the interrupting source. The corresponding EIR bit reflects the state of the interrupt signal even if the corresponding EIMR bit is cleared. 0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
18–0	—	Reserved, should be cleared.

32.6.4.3 Receive Descriptor Active Register (RDAR)

RDAR is a command register, written to by the user, which indicates that the receive descriptor ring has been updated (empty receive buffers have been produced by the driver with the empty bit set).

The RDAR bit is set whenever the register is written, independent of the data actually written by the user. When set, the FEC polls the receive descriptor ring and process receive frames (provided ECR[ETHER_EN] is also set). Once the FEC polls a receive descriptor whose empty bit is not set, then the FEC will clear the RDAR bit and cease receive descriptor ring polling until the user sets the bit again, signifying that additional descriptors have been placed into the receive descriptor ring.

The RDAR is cleared at reset, and when ECR[ETHER_EN] is cleared.

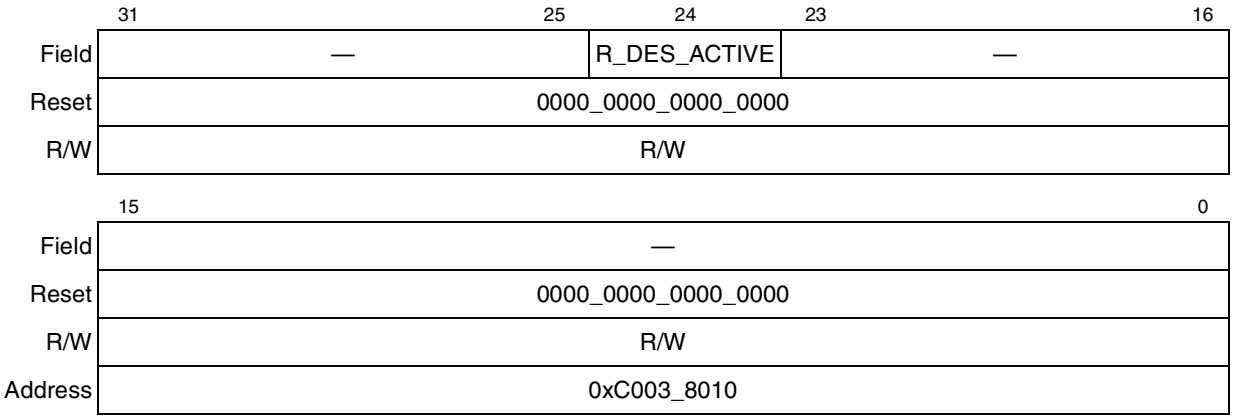


Figure 32-6. Receive Descriptor Active Register (RDAR)

Table 32-15. RDAR Field Descriptions

Bits	Name	Description
31–25	—	Reserved, should be cleared.
24	R_DES_ACTIVE	Set to one when this register is written, regardless of the value written. Cleared by the FEC device whenever no additional “empty” descriptors remain in the receive ring. Also cleared when ECR[ETHER_EN] is cleared.
23–0	—	Reserved, should be cleared.

32.6.4.4 Transmit Descriptor Active Register (TDAR)

The TDAR is a command register, written to by the user, to indicate that the transmit descriptor ring has been updated (transmit buffers have been produced by the driver with the ready bit set in the buffer descriptor).

Whenever the register is written the TDAR bit is set, independent of the data actually written by the user. When set, the FEC will poll the transmit descriptor ring and process transmit frames (provided ECR[ETHER_EN] is also set). Once the FEC polls a transmit descriptor whose ready bit is not set[, then the FEC will clear the TDAR bit and cease transmit descriptor ring polling until the user sets the bit again, signifying additional descriptors have been placed into the transmit descriptor ring.

The TDAR is cleared at reset, when ECR[ETHER_EN] is cleared, or when ECR[RESET] is set.

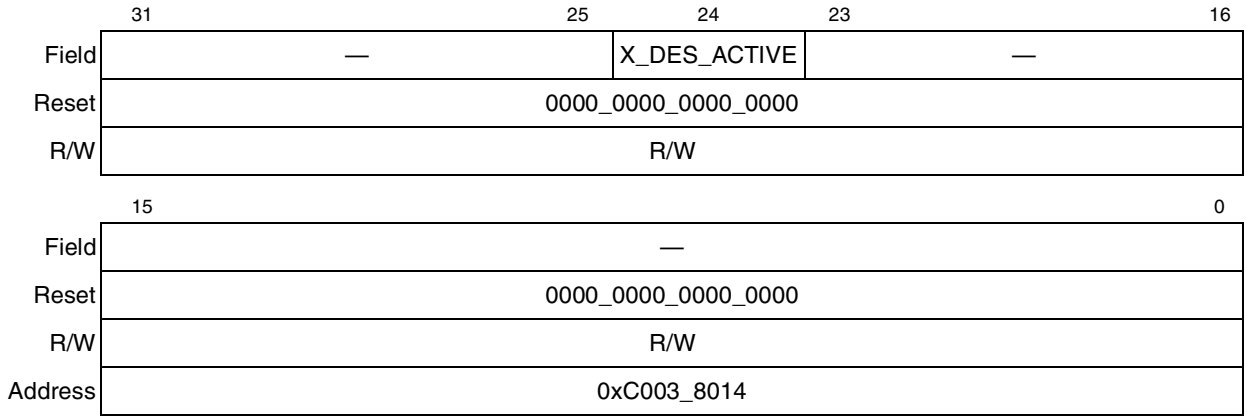


Figure 32-7. Transmit Descriptor Active Register (TDAR)

Table 32-16. TDAR Field Descriptions

Bits	Name	Description
31–25	—	Reserved, should be cleared.
24	X_DES_ACTIVE	Set to one when this register is written, regardless of the value written. Cleared by the FEC device whenever no additional “ready” descriptors remain in the transmit ring. Also cleared when ECR[ETHER_EN] is cleared.
23–0	—	Reserved, should be cleared.

32.6.4.5 Ethernet Control Register (ECR)

The ECR is used to enable/disable the FEC. ECR is a read/write user register, though both fields in this register may be altered by hardware as well.

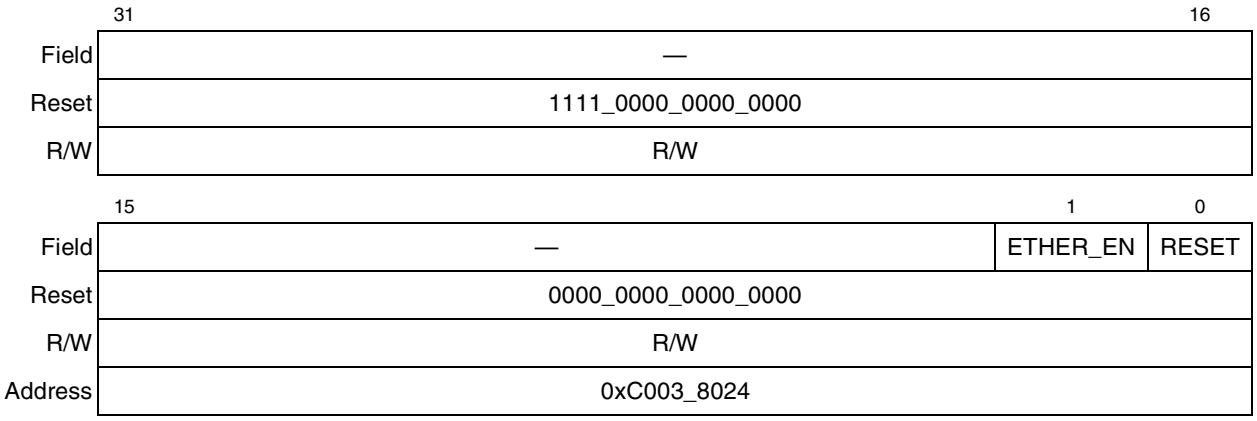


Figure 32-8. Ethernet Control Register (ECR)

Table 32-17. ECR Field Descriptions

Bits	Name	Description
31–2	—	Reserved.
1	ETHER_EN	When this bit is set, the FEC is enabled, and reception and transmission are possible. When this bit is cleared, reception is immediately stopped and transmission is stopped after a bad CRC is appended to any currently transmitted frame. The buffer descriptor(s) for an aborted transmit frame are not updated after clearing this bit. When ETHER_EN is deasserted, the DMA, buffer descriptor, and FIFO control logic are reset, including the buffer descriptor and FIFO pointers. The ETHER_EN bit is altered by hardware under the following conditions: • ECR[RESET] is set by software, in which case ETHER_EN will be cleared • an error condition causes the EIR[EBERR] bit to set, in which case ETHER_EN will be cleared
0	RESET	When this bit is set, the equivalent of a hardware reset is performed but it is local to the FEC. ETHER_EN is cleared and all other FEC registers take their reset values. Also, any transmission/reception currently in progress is abruptly aborted. This bit is automatically cleared by hardware during the reset sequence. The reset sequence takes approximately 8 system clock cycles after RESET is written with a 1.

32.6.4.6 MII Management Frame Register (MMFR)

The MMFR is accessed by the user and does not reset to a defined value. The MMFR is used to communicate with the attached MII compatible PHY device(s), providing read/write access to their MII registers. Performing a write to MMFR will cause a management frame to be sourced unless the MII-SPEED field of the MSCR has been set to 0, in which case MSCR is set to a non-zero value and an MII frame is generated with the data previously written to MMFR. This allows MMFR and MSCR to be programmed in either order if the MII-SPEED field of MSCR is currently zero (for further details on MSCR, see Section 32.6.4.7, MII Speed Control Register (MSCR)").

	31 30	29 28	27 23	22 18	17 16
Field	ST	OP	PA	RA	TA
Reset	Undefined				
R/W	R/W				

	15 0
Field	DATA
Reset	Undefined
R/W	R/W
Address	0xC003_8040

Figure 32-9. MII Management Frame Register (MMFR)

Table 32-18. MMFR Field Descriptions

Bit	Name	Description
31–30	ST	Start of frame delimiter. These bits must be programmed to 01 for a valid MII management frame.
29–28	OP	Operation code. This field must be programmed to 10 (read) or 01 (write) to generate a valid MII management frame. A value of 11 will produce “read” frame operation while a value of 00 will produce “write” frame operation, but these frames will not be MII compliant.
27–23	PA	PHY address. This field specifies one of up to 32 attached PHY devices.
22–18	RA	Register address. This field specifies one of up to 32 registers within the specified PHY device.
17–16	TA	Turn around. This field must be programmed to 10 to generate a valid MII management frame.
15–0	DATA	Management frame data. This is the field for data to be written to or read from the PHY register.

To perform a read or write operation on the MII Management Interface, the MMFR must be written to by the user. To generate a valid read or write management frame, the ST field must be written with a 01 pattern, and the TA field must be written with a 10. If other patterns are written to these fields, a frame will be generated but will not comply with the IEEE 802.3 MII definition.

To generate an IEEE 802.3-compliant MII Management Interface write frame (write to a PHY register), the user must write {01 01 PHYAD REGAD 10 DATA} to the bit fields of the MMFR, as shown in Table 32-18. Writing this pattern will cause the control logic to shift out the data in the MMFR following a preamble generated by the control state machine. During this time the contents of the MMFR will be altered as the contents are serially shifted and will be unpredictable if read by the user. Once the write management frame operation has completed, the MII interrupt will be generated. At this time the contents of the MMFR will match the original value written.

To generate an MII Management Interface read frame (read a PHY register) the user must write {01 10 PHYAD REGAD 10 XXXX} to the bit fields of the MMFR shown in Table 32-18 (the contents of the 4-bit DATA field are arbitrary). Writing this pattern will cause the control logic to shift out the data in the MMFR following a preamble generated by the control state machine. During this time the contents of the MMFR will be altered as the contents are serially shifted, and will be unpredictable if read by the user. Once the read management frame operation has completed, the MII interrupt will be generated. At this time the contents of the MMFR will match the original value written except for the DATA field whose contents have been replaced by the value read from the PHY register.

If the MMFR is written to while frame generation is in progress, the frame contents will be altered. Software should use the MII interrupt to avoid writing to the MMFR while frame generation is in progress.

32.6.4.7 MII Speed Control Register (MSCR)

The MSCR provides control of the MII clock (FEC_MDC pin) frequency, and allows a preamble drop on the MII management frame.

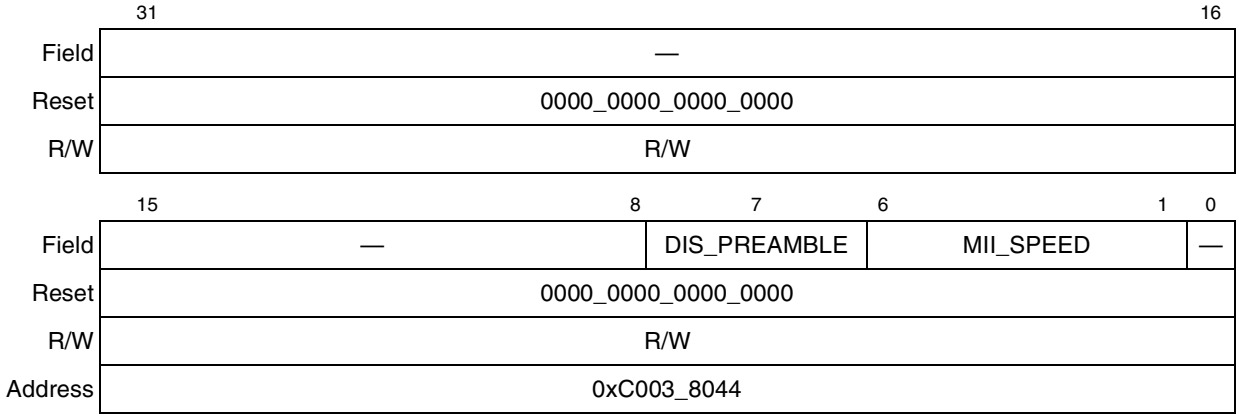


Figure 32-10. MII Speed Control Register (MSCR)

Table 32-19. MSCR Field Descriptions

Bits	Name	Description
31–8	—	Reserved, should be cleared.
7	DIS_PREAMBLE	Asserting this bit will cause preamble (32 1's) not to be prepended to the MII management frame. The MII standard allows the preamble to be dropped if the attached PHY device(s) does not require it.
6–1	MII_SPEED	MII_SPEED controls the frequency of the MII management interface clock (FEC_MDC) relative to the system clock. A value of 0 in this field will "turn off" the FEC_MDC and leave it in low voltage state. Any non-zero value will result in the FEC_MDC frequency of 1/(MII_SPEED*2) of the system clock frequency.
0	—	Reserved, should be cleared.

The MII_SPEED field must be programmed with a value to provide an FEC_MDC frequency of less than or equal to 2.5 MHz to be compliant with the IEEE 802.3 MII specification. The MII_SPEED must be set to a non-zero value in order to source a read or write management frame. After the management frame is complete the MSCR may optionally be set to zero to turn off the FEC_MDC. The FEC_MDC generated will have a 50% duty cycle except when MII_SPEED is changed during operation (change will take effect following either a rising or falling edge of FEC_MDC).

The FEC_MDC frequency depends on both the system clock frequency and the MII_SPEED register. If the system clock is 25 MHz, programming the MII_SPEED register to 0x0000_0005 results in an

FEC_MDC frequency of 25 MHz × 1/10 = 2.5 MHz. Table 32-20 shows the optimum values for MII_SPEED for different system clock frequencies.

Table 32-20. Programming Examples for MSCR

System Clock Frequency	MII_SPEED (field in reg)	FEC_MDC frequency
25 MHz	0x5	2.5 MHz
33 MHz	0x7	2.36 MHz
40 MHz	0x8	2.5 MHz
50 MHz	0xA	2.5 MHz
66 MHz	0xD	2.54 MHz

32.6.4.8 MIB Control Register (MIBC)

The MIB control register is a read/write register used to provide control of and to observe the state of the Message Information Block (MIB). This register is accessed by user software if there is a need to disable the MIB operation. For example, in order to clear all MIB counters in RAM the user should disable the MIB, then clear all the MIB RAM locations, then enable the MIB. The MIB_DISABLE bit is reset to 1. See Table 32-11 for the locations of the MIB counters.

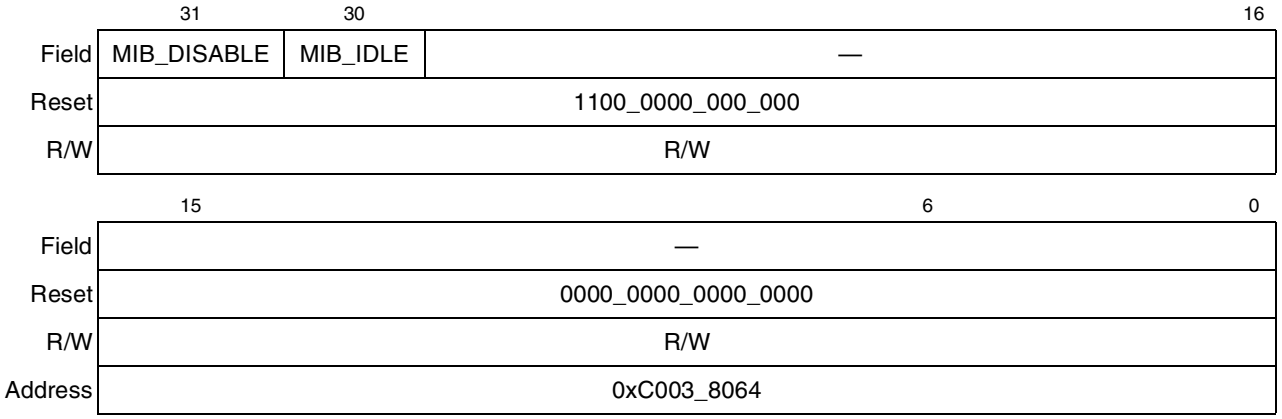


Figure 32-11. MIB Control Register (MIBC)

Table 32-21. MIBC Field Descriptions

Bits	Name	Description
31	MIB_DISABLE	A read/write control bit. If set, the MIB logic will halt and not update any MIB counters.
30	MB_IDLE	A read-only status bit. If set the MIB block is not currently updating any MIB counters.
29–0	—	Reserved.

32.6.4.9 Receive Control Register (RCR)

The RCR is programmed by the user, and controls the operational mode of the receive block. It should be written to only when ECR[ETHER_EN] = 0 (initialization time).

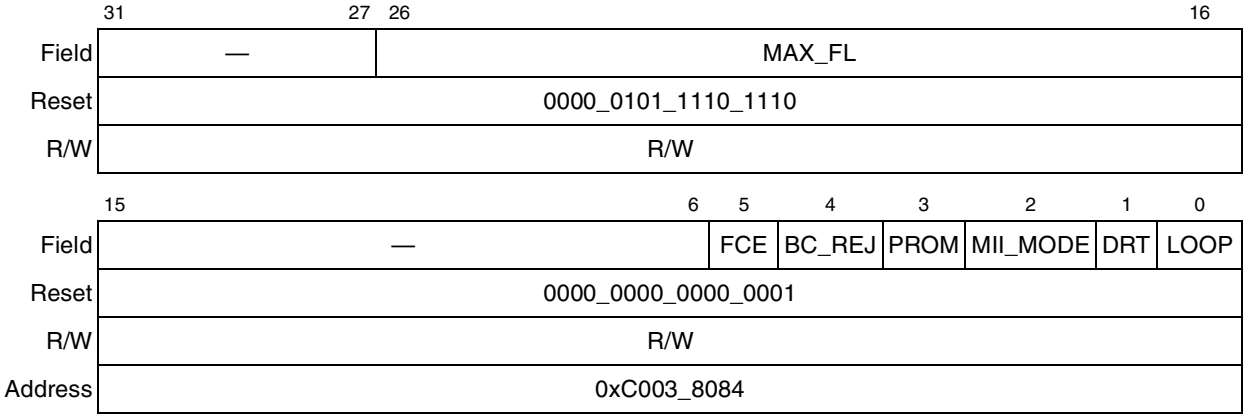


Figure 32-12. Receive Control Register (RCR)

Table 32-22. RCR Field Descriptions

Bits	Name	Description
31–27	—	Reserved, should be cleared.
26–16	MAX_FL	Maximum frame length. Resets to decimal 1518. Length is measured starting at DA and includes the CRC at the end of the frame. Transmit frames longer than MAX_FL will cause the BABT interrupt to occur. Receive Frames longer than MAX_FL will cause the BABR interrupt to occur and will set the LG bit in the end of frame receive buffer descriptor. The recommended default value to be programmed by the user is 1518 or 1522 (if VLAN Tags are supported).
15–6	—	Reserved, should be cleared.
5	FCE	Flow control enable. If asserted, the receiver will detect PAUSE frames. Upon PAUSE frame detection, the transmitter will stop transmitting data frames for a given duration.
4	BC_REJ	Broadcast frame reject. If asserted, frames with DA (destination address) = FF_FF_FF_FF_FF_FF will be rejected unless the PROM bit is set. If both BC_REJ and PROM = 1, then frames with broadcast DA will be accepted and the M (MISS) bit will be set in the receive buffer descriptor.
3	PROM	Promiscuous mode. All frames are accepted regardless of address matching.
2	MII_MODE	Media independent interface mode. Selects external interface mode. Setting this bit to one selects MII mode, setting this bit equal to zero selects 7-wire mode (used only for serial 10 Mbps). This bit controls the interface mode for both transmit and receive blocks.
1	DRT	Disable receive on transmit. 0 Receive path operates independently of transmit (use for full duplex or to monitor transmit activity in half duplex mode). 1 Disable reception of frames while transmitting (normally used for half duplex mode).
0	LOOP	Internal loopback. If set, transmitted frames are looped back internal to the device and the transmit output signals are not asserted. The system clock is substituted for the FEC_TX_CLK when LOOP is asserted. DRT must be set to zero when asserting LOOP.

32.6.4.10 Transmit Control Register (TCR)

This register is read/write, and is written by the user to configure the transmit block. This register is cleared at system reset. Bits 2 and 1 should be modified only when ECR[ETHER_EN] = 0.

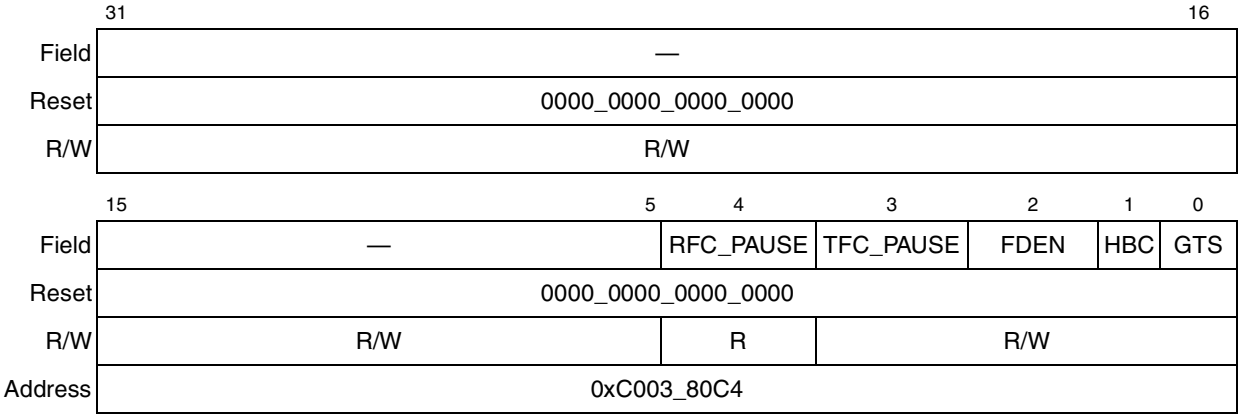


Figure 32-13. Transmit Control Register (TCR)

Table 32-23. TCR Field Descriptions

Bits	Name	Description
31–5	—	Reserved, should be cleared.
4	RFC_PAUSE	Receive frame control pause. This read-only status bit will be asserted when a full duplex flow control pause frame has been received and the transmitter is paused for the duration defined in this pause frame. This bit will automatically clear when the pause duration is complete.
3	TFC_PAUSE	Transmit frame control pause. Transmits a PAUSE frame when asserted. When this bit is set, the FEC will stop transmission of data frames after the current transmission is complete. At this time, the GRA interrupt in the EIR register will be asserted. With transmission of data frames stopped, the FEC will transmit a MAC Control PAUSE frame. Next, the FEC will clear the TFC_PAUSE bit and resume transmitting data frames. Note that if the transmitter is paused due to user assertion of GTS or reception of a PAUSE frame, the FEC may still transmit a MAC Control PAUSE frame.
2	FDEN	Full duplex enable. If set, frames are transmitted independent of carrier sense and collision inputs. This bit should only be modified when ETHER_EN is deasserted.
1	HBC	Heartbeat control. If set, the heartbeat check is performed following end of transmission and the HB bit in the status register will be set if the collision input does not assert within the heartbeat window. This bit should only be modified when ETHER_EN is deasserted.
0	GTS	Graceful transmit stop. When this bit is set, the FEC will stop transmission after any frame that is currently being transmitted is complete and the GRA interrupt in the EIR register will be asserted. If frame transmission is not currently underway, the GRA interrupt will be asserted immediately. Once transmission has completed, a “restart” can be accomplished by clearing the GTS bit. The next frame in the transmit FIFO will then be transmitted. If an early collision occurs during transmission when GTS = 1, transmission will stop after the collision. The frame will be transmitted again once GTS is cleared. Note that there may be old frames in the transmit FIFO that will be transmitted when GTS is reasserted. To avoid this deassert ECR[ETHER_EN] following the GRA interrupt.

32.6.4.11 Physical Address Low Register (PALR)

The PALR is written by the user. This register contains the lower 32 bits (bytes 0,1,2,3) of the 48-bit address used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. This register is also used for bytes 0 through 3 of the 6-byte Source Address field when transmitting PAUSE frames. This register is not reset and must be initialized by the user.

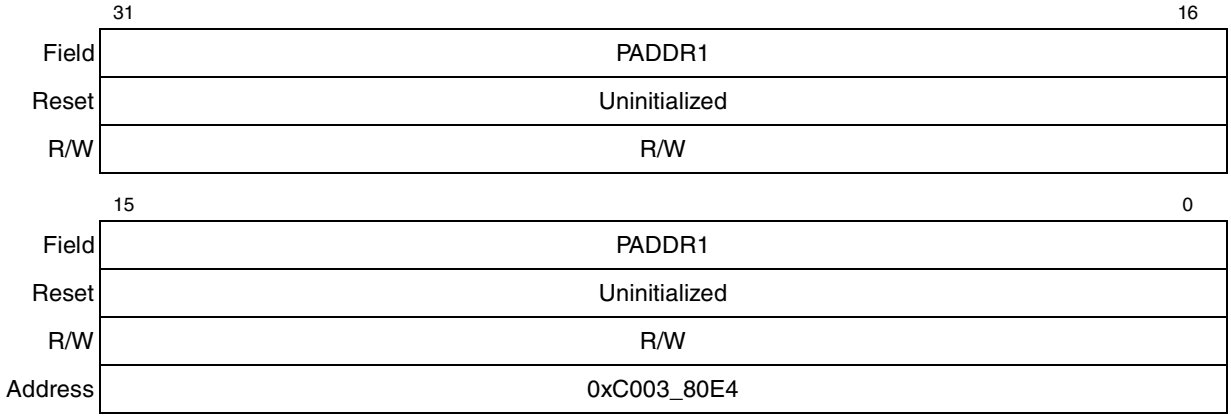


Figure 32-14. Physical Address Low Register (PALR)

Table 32-24. PALR Field Descriptions

Bits	Name	Description
31–0	PADDR1	Bytes 0 (bits 31–24), 1 (bits 23–16), 2 (bits 15–8) and 3 (bits 7–0) of the 6-byte individual address to be used for exact match, and the Source Address field in PAUSE frames.

32.6.4.12 Physical Address High Register (PAUR)

The PAUR is written by the user. This register contains the upper 16 bits (bytes 4 and 5) of the 48-bit address used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. In addition, this register is used in bytes 4 and 5 of the 6-byte Source Address field when transmitting PAUSE frames. Bits 15–0 of PAUR contain a constant type field (0x8808)

used for transmission of PAUSE frames. This register is not reset, and bits 31–16 must be initialized by the user.

	31	16
Field	PADDR2	
Reset	Uninitialized	
R/W	R/W	

	15	0
Field	TYPE	
Reset	1000_1000_0000_1000	
R/W	R	
Address	0xC003_80E8	

Figure 32-15. Physical Address High Register (PAUR)

Table 32-25. PAUR Field Descriptions

BIts	Name	Description
31–16	PADDR2	Bytes 4 (bits 31:24) and 5 (bits 23:16) of the 6-byte individual address to be used for exact match, and the Source Address field in PAUSE frames.
15–0	TYPE	Type field in PAUSE frames. These 16-bits are a constant value of 0x8808.

32.6.4.13 Opcode/Pause Duration Register (OPD)

The OPD register is read/write accessible. This register contains the 16-bit Opcode, and 16-bit pause duration fields used in transmission of a PAUSE frame. The Opcode field is a constant value, 0x0001. When another node detects a PAUSE frame, that node will pause transmission for the duration specified in the pause duration field. This register is not reset and must be initialized by the user.

	31	16
Field	OPCODE	
Reset	0000_0000_0000_0001	
R/W	R	

	15	0
Field	PAUSE_DUR	
Reset	Uninitialized	
R/W	R/W	
Address	0xC003_80EC	

Figure 32-16. Opcode/Pause Duration Register (OPD)

Table 32-26. OPD Field Descriptions

Bits	Name	Description
31–16	OPCODE	Opcode field used in PAUSE frames. These bits are a constant, 0x0001.
15–0	PAUSE_DUR	Pause Duration field used in PAUSE frames.

32.6.4.14 Descriptor Individual Upper Address Register (IAUR)

The IAUR is written by the user. This register contains the upper 32 bits of the 64-bit individual address hash table used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. This register is not reset and must be initialized by the user.

	31	16
Field	IADDR1	
Reset	Uninitialized	
R/W	R/W	

	15	0
Field	IADDR1	
Reset	Uninitialized	
R/W	R/W	
Address	0xC003_8118	

Figure 32-17. Descriptor Individual Upper Address Register (IAUR)

Table 32-27. IAUR Field Descriptions

Bits	Name	Description
31–0	IADDR1	The upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR1 contains hash index bit 63. Bit 0 of IADDR1 contains hash index bit 32.

32.6.4.15 Descriptor Individual Lower Address Register (IALR)

The IALR is written by the user. This register contains the lower 32 bits of the 64-bit individual address hash table used in the address recognition process to check for possible match with the DA field of receive frames with an individual DA. This register is not reset and must be initialized by the user.

	31	16
Field	IADDR2	
Reset	Uninitialized	
R/W	R/W	

	15	0
Field	IADDR2	
Reset	Uninitialized	
R/W	R/W	
Address	0xC003_811C	

Figure 32-18. Descriptor Individual Lower Address Register (IALR)

Table 32-28. IALR Field Descriptions

Bits	Name	Description
31–0	IADDR2	The lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR2 contains hash index bit 31. Bit 0 of IADDR2 contains hash index bit 0.

32.6.4.16 Descriptor Group Upper Address Register (GAUR)

The GAUR is written by the user. This register contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. This register must be initialized by the user.

	31	16
Field	GADDR1	
Reset	Uninitialized	
R/W	R/W	

	15	0
Field	GADDR1	
Reset	Uninitialized	
R/W	R/W	
Address	0xC003_8120	

Figure 32-19. Descriptor Group Upper Address Register (GAUR)

Table 32-29. GAUR Field Descriptions

Bits	Name	Description
31–0	GADDR1	The GADDR1 register contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR1 contains hash index bit 63. Bit 0 of GADDR1 contains hash index bit 32.

32.6.4.17 Descriptor Group Lower Address Register (GALR)

The GALR is written by the user. This register contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. This register must be initialized by the user.

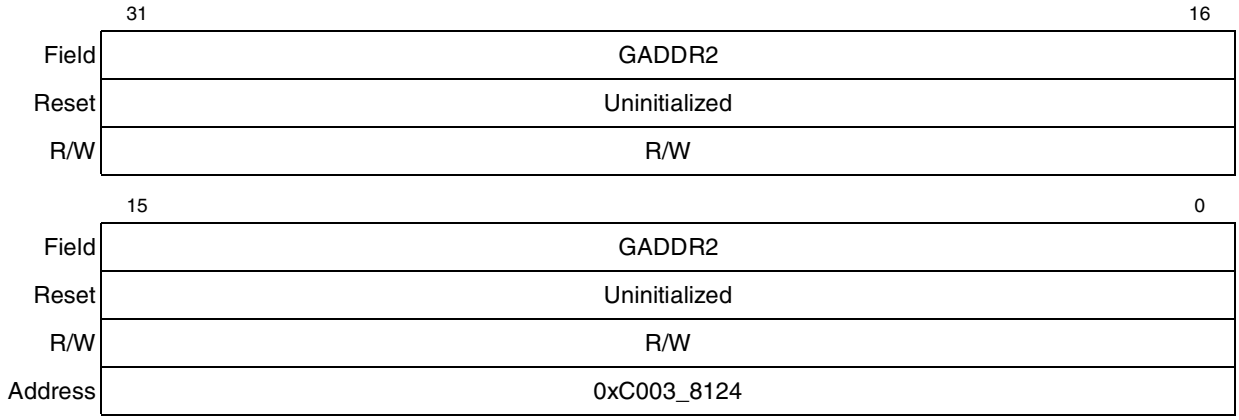


Figure 32-20. Descriptor Group Lower Address Register (GALR)

Table 32-30. GALR Field Descriptions

Bits	Name	Description
31–0	GADDR2	The GADDR2 register contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR2 contains hash index bit 31. Bit 0 of GADDR2 contains hash index bit 0.

32.6.4.18 Transmit FIFO Watermark Register (TFWR)

The TFWR is a 2-bit read/write register programmed by the user to control the amount of data required in the transmit FIFO before transmission of a frame can begin. This allows the user to minimize transmit latency (TFWR = 0x) or allow for larger bus access latency (TFWR = 11) due to contention for the system bus. Setting the watermark to a high value will minimize the risk of transmit FIFO underrun due to

contention for the system bus. The byte counts associated with the TFWR field may need to be modified to match a given system requirement (worst case bus access latency by the transmit data DMA channel).

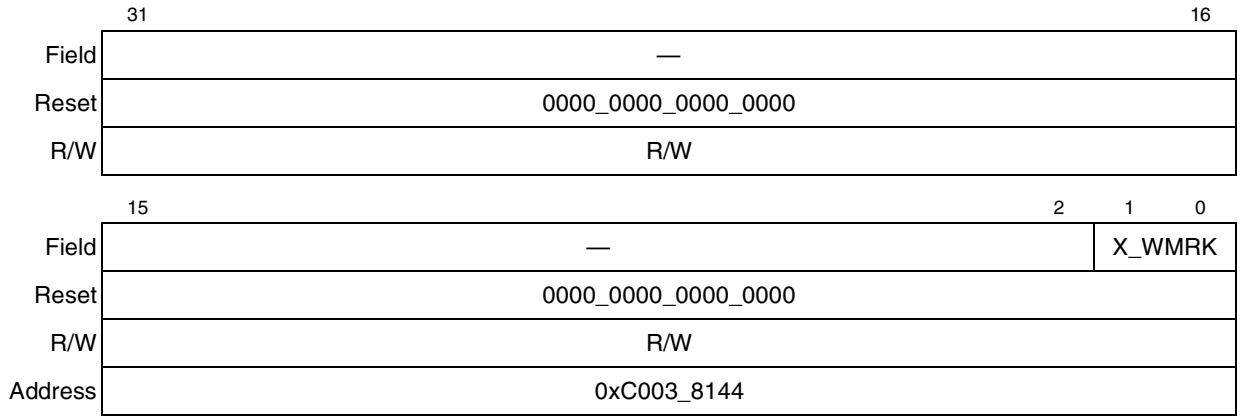


Figure 32-21. FIFO Transmit FIFO Watermark Register (TFWR)

Table 32-31. TFWR Field Descriptions

Bits	Name	Description
31–2	—	Reserved, should be cleared.
1–0	X_WMRK	Number of bytes written to transmit FIFO before transmission of a frame begins 0x 64 bytes written 10 128 bytes written 11 192 bytes written

32.6.4.19 FIFO Receive Bound Register (FRBR)

The FRBR is an 8-bit register that the user can read to determine the upper address bound of the FIFO RAM. Drivers can use this value, along with the FRSR to appropriately divide the available FIFO RAM between the transmit and receive data paths.

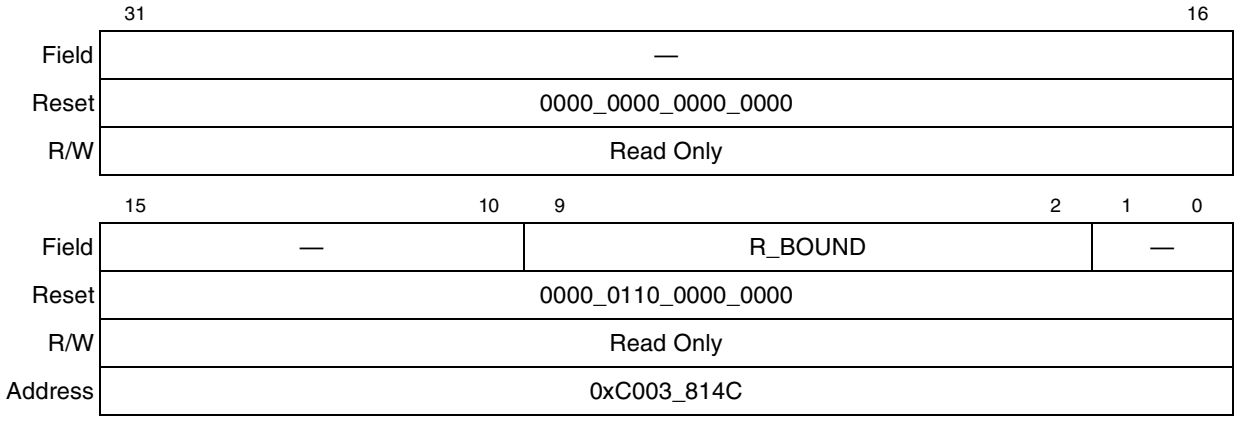


Figure 32-22. FIFO Receive Bound Register (FRBR)

Table 32-32. FRBR Field Descriptions

Bits	Name	Description
31–10	—	Reserved, read as 0 (except bit 10, which is read as 1).
9–2	R_BOUND	Read-only. Highest valid FIFO RAM address.
1–0	—	Reserved, should be cleared.

32.6.4.20 FIFO Receive Start Register (FRSR)

The FRSR, shown in Figure 32-23 and Table 32-33, is an 8-bit register programmed by the user to indicate the starting address of the receive FIFO. FRSR marks the boundary between the transmit and receive FIFOs. The transmit FIFO uses addresses from the start of the FIFO to the location four bytes before the address programmed into the FRSR. The receive FIFO uses addresses from FRSR to FRBR inclusive.

The FRSR is initialized by hardware at reset. FRSR only needs to be written to change the default value.

	31															16
Field	—															
Reset	0000_0000_0000_0000															
R/W	R/W															

	15					10	9							2	1	0
Field	—						R_FSTART								—	
Reset	0000_0101_0000_0000															
R/W	R/W															
Address	0xC003_8150															

Figure 32-23. FIFO Receive Start Register (FRSR)

Table 32-33. FRSR Field Descriptions

Bits	Name	Description
31–10	—	Reserved, read as 0 (except bit 10, which is read as 1).
9–2	R_FSTART	Address of first receive FIFO location. Acts as delimiter between receive and transmit FIFOs.
1–0	—	Reserved, read as 0.

32.6.4.21 Receive Buffer Descriptor Ring Start Register (ERDSR)

The ERDSR, shown in Figure 32-24 and Table 32-34, is written by the user. It provides a pointer to the start of the circular receive buffer descriptor queue in external memory. This pointer must be 32-bit aligned; however, it is recommended it be made 128-bit aligned (evenly divisible by 16).

This register is not reset and must be initialized by the user prior to operation.

	31	16
Field	R_DES_START	
Reset	Uninitialized	
R/W	R/W	

	15	2	1	0
Field	R_DES_START		—	
Reset	Uninitialized			
R/W	R/W			
Address	0xC003_8180			

Figure 32-24. Receive Descriptor Ring Start Register (ERDSR)

Table 32-34. ERDSR Field Descriptions

Bits	Name	Description
31–2	R_DES_START	Pointer to start of receive buffer descriptor queue.
1–0	—	Reserved, should be cleared.

32.6.4.22 Transmit Buffer Descriptor Ring Start Register (ETDSR)

The ETDSR, shown in Figure 32-25 and Table 32-35, is written by the user. It provides a pointer to the start of the circular transmit buffer descriptor queue in external memory. This pointer must be 32-bit

aligned; however, it is recommended it be made 128-bit aligned (evenly divisible by 16). Bits 1 and 0 should be written to 0 by the user. Non-zero values in these two bit positions are ignored by the hardware.

This register is not reset and must be initialized by the user prior to operation.

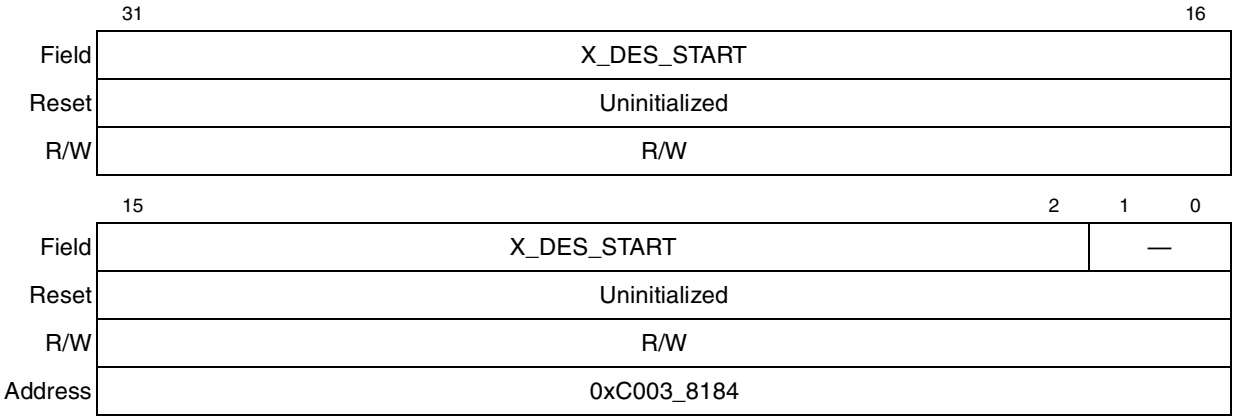


Figure 32-25. Transmit Buffer Descriptor Ring Start Register (ETDSR)

Table 32-35. ETDSR Field Descriptions

Bits	Name	Description
31–2	X_DES_START	Pointer to start of transmit buffer descriptor queue.
1–0	—	Reserved, should be cleared.

32.6.4.23 Receive Buffer Size Register (EMRBR)

The EMRBR, shown in Figure 32-26 and Table 32-36, is a 9-bit register programmed by the user. The EMRBR dictates the maximum size of all receive buffers. Note that because receive frames will be truncated at 2k-1(2047) bytes, bits 31-11 are not used. The programmed value should account for the fact that the receive CRC is always written into the last receive buffer. To allow one maximum size frame per buffer, EMRBR must be set to RCR[MAX_FL] or larger. The EMRBR must be evenly divisible by 16. To

ensure this, bits 3-0 are forced low, and hence only bits 10-4 are actually used. To minimize bus utilization (descriptor fetches) it is recommended that EMRBR be greater than or equal to 256 bytes.

The EMRBR does not reset, and must be initialized by the user.

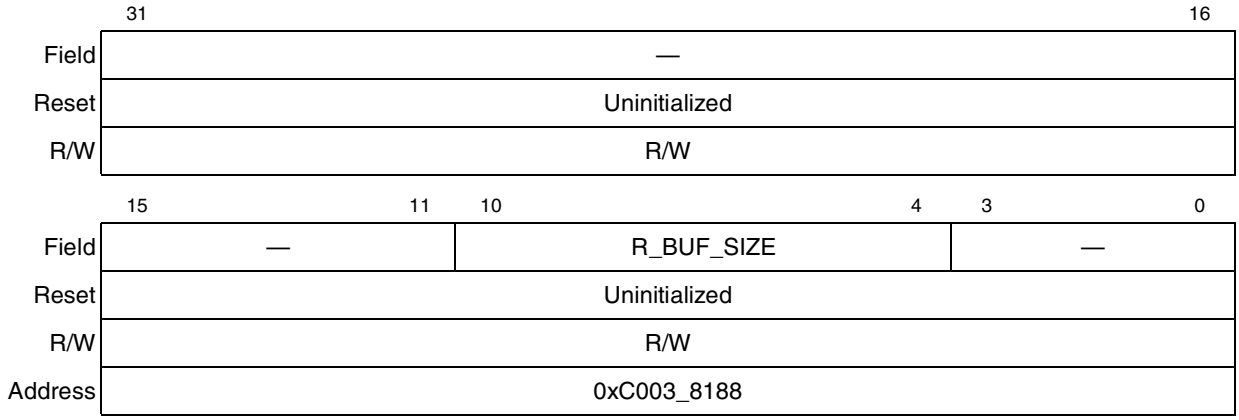


Figure 32-26. Receive Buffer Size Register (EMRBR)

Table 32-36. EMRBR Field Descriptions

Bits	Name	Description
30–11	—	Reserved, should be written to 0 by the host processor.
10–4	R_BUF_SIZE	Receive buffer size.
3–0	—	Reserved, should be written to 0 by the host processor.

32.6.5 Buffer Descriptors

This section provides a description of the operation of the driver/DMA via the buffer descriptors. It is followed by a detailed description of the receive and transmit descriptor fields.

32.6.5.1 Driver/DMA Operation with Buffer Descriptors

The data for the FEC frames must reside in memory external to the FEC. The data for a frame is placed in one or more buffers. Associated with each buffer is a buffer descriptor (BD) which contains a starting address (pointer), data length, and status/control information (which contains the current state for the buffer). To permit maximum user flexibility, the BDs are also located in external memory and are read in by the FEC DMA engine.

Software “produces” buffers by allocating/initializing memory and initializing buffer descriptors. Setting the RxBD[E] or TxBD[R] bit “produces” the buffer. Software writing to either the TDAR or RDAR tells the FEC that a buffer has been placed in external memory for the transmit or receive data traffic, respectively. The hardware reads the BDs and “consumes” the buffers after they have been produced. After the data DMA is complete and the buffer descriptor status bits have been written by the DMA engine, the RxBD[E] or TxBD[R] bit will be cleared by hardware to signal the buffer has been “consumed.” Software may poll the BDs to detect when the buffers have been consumed or may rely on the buffer/frame interrupts. These buffers may then be processed by the driver and returned to the free list.

The ECR[ETHER_EN] signal operates as a reset to the BD/DMA logic. When ECR[ETHER_EN] is deasserted the DMA engine BD pointers are reset to point to the starting transmit and receive BDs. The buffer descriptors are not initialized by hardware during reset. At least one transmit and receive buffer descriptor must be initialized by software before the ECR[ETHER_EN] bit is set.

The buffer descriptors operate as two separate rings. ERDSR defines the starting address for receive BDs and ETDSR defines the starting address for transmit BDs. The last buffer descriptor in each ring is defined by the Wrap (W) bit. When set, W indicates that the next descriptor in the ring is at the location pointed to by ERDSR and ETDSR for the receive and transmit rings, respectively.

NOTE

Buffer descriptor rings must start on a 128-bit boundary.

32.6.5.1.1 Driver/DMA Operation with Transmit BDs

Typically a transmit frame will be divided between multiple buffers. An example is to have an application payload in one buffer, TCP header in a 2nd buffer, IP header in a 3rd buffer, Ethernet/IEEE 802.3 header in a 4th buffer. The FEC does not prepend the Ethernet header (Destination Address, Source Address, Length/Type field(s)), so this must be provided by the driver in one of the transmit buffers. The FEC can append the Ethernet CRC to the frame. Whether the CRC is appended by the FEC or by the driver is determined by the TC bit in the transmit BD which must be set by the driver.

The driver (TxBD software producer) should set up Tx BDs in such a way that a complete transmit frame is given to the hardware at once. If a transmit frame consists of three buffers, the BDs should be initialized with pointer, length and control (W, L, TC, ABC) and then the TxBD[R] bits should be set = 1 in reverse order (3rd, 2nd, 1st BD) to insure that the complete frame is ready in memory before the DMA begins. If the TxBDs are set up in order, the DMA Controller could DMA the first BD before the 2nd was made available, potentially causing a transmit FIFO underrun.

In the FEC, the DMA is notified by the driver that new transmit frame(s) are available by writing to the TDAR register. When this register is written to (data value is not significant) the FEC RISC will tell the DMA to read the next transmit BD in the ring. Once started, the RISC + DMA will continue to read and interpret transmit BDs in order and DMA the associated buffers, until a transmit BD is encountered with the R bit = 0. At this point the FEC will poll this BD one more time. If the R bit = 0 the second time, then the RISC will stop the transmit descriptor read process until software sets up another transmit frame and writes to TDAR.

When the DMA of each transmit buffer is complete, the DMA writes back to the BD to clear the R bit, indicating that the hardware consumer is finished with the buffer.

32.6.5.1.2 Driver/DMA Operation with Receive BDs

Unlike the transmit case, the length of the receive frame is unknown by the driver ahead of time. Therefore the driver must set a variable to define the length of all receive buffers. In the FEC, this variable is written to the EMRBR register.

The driver (RxBD software producer) should set up some number of “empty” buffers for the Ethernet by initializing the address field and the E and W bits of the associated receive BDs. The hardware (receive

DMA) will consume these buffers by filling them with data as frames are received and clearing the E bit and writing to the L (1 indicates last buffer in frame) bit, the frame status bits (if L = 1) and the length field.

If a receive frame spans multiple receive buffers, the L bit is only set for the last buffer in the frame. For non-last buffers, the length field in the receive BD will be written by the DMA (at the same time the E bit is cleared) with the default receive buffer length value. For end of frame buffers the receive BD will be written with L = 1 and information written to the status bits (M, BC, MC, LG, NO, CR, OV, TR). Some of the status bits are error indicators which, if set, indicate the receive frame should be discarded and not given to higher layers. The frame status/length information is written into the receive FIFO following the end of the frame (as a single 32-bit word) by the receive logic. The length field for the end of frame buffer will be written with the length of the entire frame, not just the length of the last buffer.

For simplicity the driver may assign the default receive buffer length to be large enough to contain an entire frame, keeping in mind that a malfunction on the network or out-of-specification implementation could result in giant frames. Frames of 2k (2048) bytes or larger are truncated by the FEC at 2047 bytes so software never sees a receive frame larger than 2047 bytes.

Similar to transmit, the FEC will poll the receive descriptor ring after the driver sets up receive BDs and writes to the RDAR register. As frames are received the FEC will fill receive buffers and update the associated BDs, then read the next BD in the receive descriptor ring. If the FEC reads a receive BD and finds the E bit = 0, it will poll this BD once more. If the BD = 0 a second time the FEC will stop reading receive BDs until the driver writes to RDAR.

32.6.5.2 Ethernet Receive Buffer Descriptor (RxBD)

In the RxBD, the user initializes the E and W bits in the first longword and the pointer in second longword. When the buffer has been DMA'd, the Ethernet controller will modify the E, L, M, BC, MC, LG, NO, CR, OV, and TR bits and write the length of the used portion of the buffer in the first longword. The M, BC,

MC, LG, NO, CR, OV and TR bits in the first longword of the buffer descriptor are only modified by the Ethernet controller when the L bit is set.

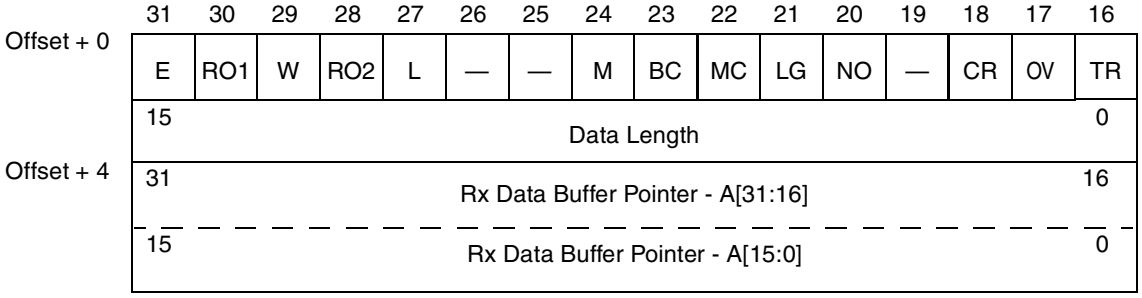


Figure 32-27. Receive Buffer Descriptor (RxBD)

Table 32-37. Receive Buffer Descriptor Field Definitions

Word	Location	Field Name	Description
Offset + 0	Bit 31	E	Empty. Written by the FEC (=0) and user (=1). 0 The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The status and length fields have been updated as required. 1 The data buffer associated with this BD is empty, or reception is currently in progress.
Offset + 0	Bit 30	RO1	Receive software ownership. This field is reserved for use by software. This read/write bit will not be modified by hardware, nor will its value affect hardware.
Offset + 0	Bit 29	W	Wrap. Written by user. 0 The next buffer descriptor is found in the consecutive location 1 The next buffer descriptor is found at the location defined in ERDSR.
Offset + 0	Bit 28	RO2	Receive software ownership. This field is reserved for use by software. This read/write bit will not be modified by hardware, nor will its value affect hardware.
Offset + 0	Bit 27	L	Last in frame. Written by the FEC. 0 The buffer is not the last in a frame. 1 The buffer is the last in a frame.
Offset + 0	Bits 26-25	—	Reserved.
Offset + 0	Bit 24	M	Miss. Written by the FEC. This bit is set by the FEC for frames that were accepted in promiscuous mode, but were flagged as a “miss” by the internal address recognition. Thus, while in promiscuous mode, the user can use the M-bit to quickly determine whether the frame was destined to this station. This bit is valid only if the L-bit is set and the PROM bit is set. 0 The frame was received because of an address recognition hit. 1 The frame was received because of promiscuous mode.
Offset + 0	Bit 23	BC	Will be set if the DA is broadcast (FF-FF-FF-FF-FF-FF).
Offset + 0	Bit 22	MC	Will be set if the DA is multicast and not BC.

Table 32-37. Receive Buffer Descriptor Field Definitions (continued)

Word	Location	Field Name	Description
Offset + 0	Bit 21	LG	Rx frame length violation. Written by the FEC. A frame length greater than RCR[MAX_FL] was recognized. This bit is valid only if the L-bit is set. The receive data is not altered in any way unless the length exceeds 2047 bytes.
Offset + 0	Bit 20	NO	Receive non-octet aligned frame. Written by the FEC. A frame that contained a number of bits not divisible by 8 was received, and the CRC check that occurred at the preceding byte boundary generated an error. This bit is valid only if the L-bit is set. If this bit is set the CR bit will not be set.
Offset + 0	Bit 19	—	Reserved.
Offset + 0	Bit 18	CR	Receive CRC error. Written by the FEC. This frame contains a CRC error and is an integral number of octets in length. This bit is valid only if the L-bit is set.
Offset + 0	Bit 17	OV	Overrun. Written by the FEC. A receive FIFO overrun occurred during frame reception. If this bit is set, the other status bits, M, LG, NO, CR, and CL lose their normal meaning and will be zero. This bit is valid only if the L-bit is set.
Offset + 0	Bit 16	TR	Will be set if the receive frame is truncated (frame length > 2047 bytes). If the TR bit is set the frame should be discarded and the other error bits should be ignored as they may be incorrect.
Offset + 0	Bits [15:0]	Data Length	Data length. Written by the FEC. Data length is the number of octets written by the FEC into this BD's data buffer if L = 0 (the value will be equal to EMRBR), or the length of the frame including CRC if L = 1. It is written by the FEC once as the BD is closed.
Offset + 4	Bits [31:16]	A[31:16]	RX data buffer pointer, bits [31:16][1]
Offset + 4	Bits [15:0]	A[15:0]	RX data buffer pointer, bits [15:0]

[1] The receive buffer pointer, which contains the address of the associated data buffer, must always be divisible by 16. The buffer must reside in memory external to the FEC. This value is never modified by the Ethernet controller.

NOTE

Whenever the software driver sets an E bit in one or more receive descriptors, the driver should follow that with a write to RDAR.

32.6.5.3 Ethernet Transmit Buffer Descriptor (TxBD)

Data is presented to the FEC for transmission by arranging it in buffers referenced by the channel's TxBDs. The Ethernet controller confirms transmission by clearing the ready bit (R bit) when DMA of the buffer is complete. In the TxBD the user initializes the R, W, L, and TC bits and the length (in bytes) in the first longword, and the buffer pointer in the second longword.

The FEC will set the R bit = 0 in the first longword of the BD when the buffer has been DMA'd. Status bits for the buffer/frame are not included in the transmit buffer descriptors. Transmit frame status is

indicated via individual interrupt bits (error conditions) and in statistic counters in the MIB block. See Section 32.6.3, MIB Block Counters Memory Map,” for more details.

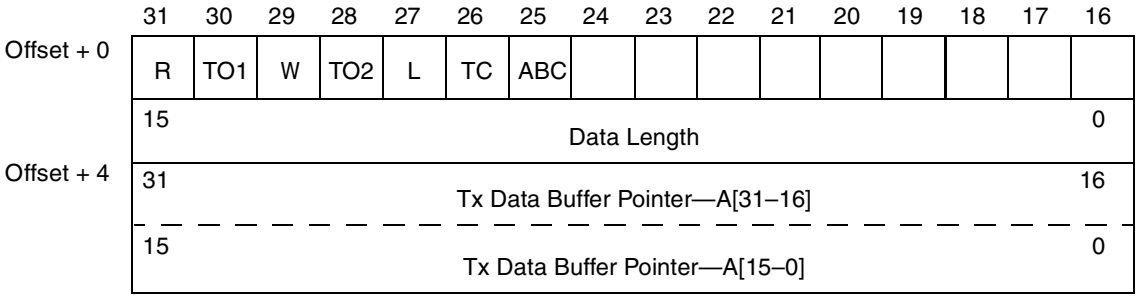


Figure 32-28. Transmit Buffer Descriptor (TxBD)

Table 32-38. Transmit Buffer Descriptor Field Definitions

Word	Location	Field Name	Description
Offset + 0	Bit 31	R	Ready. Written by the FEC and the user. 0 The data buffer associated with this BD is not ready for transmission. The user is free to manipulate this BD or its associated data buffer. The FEC clears this bit after the buffer has been transmitted or after an error condition is encountered. 1 The data buffer, which has been prepared for transmission by the user, has not been transmitted or is currently being transmitted. No fields of this BD may be written by the user once this bit is set.
Offset + 0	Bit 30	TO1	Transmit software ownership. This field is reserved for software use. This read/write bit will not be modified by hardware, nor will its value affect hardware.
Offset + 0	Bit 29	W	Wrap. Written by user. 0 The next buffer descriptor is found in the consecutive location 1 The next buffer descriptor is found at the location defined in ETDSR.
Offset + 0	BIt 28	TO2	Transmit software ownership. This field is reserved for use by software. This read/write bit will not be modified by hardware, nor will its value affect hardware.
Offset + 0	Bit 27	L	Last in frame. Written by user. 0 The buffer is not the last in the transmit frame. 1 The buffer is the last in the transmit frame.
Offset + 0	Bit 26	TC	Tx CRC. Written by user (only valid if L = 1). 0 End transmission immediately after the last data byte. 1 Transmit the CRC sequence after the last data byte.
Offset + 0	Bit 25	ABC	Append bad CRC. Written by user (only valid if L = 1). 0 No effect 1 Transmit the CRC sequence inverted after the last data byte (regardless of TC value).
Offset + 0	Bits [24:16]	—	Reserved.
Offset + 0	Bits [15:0]	Data Length	Data Length, written by user. Data length is the number of octets the FEC should transmit from this BD’s data buffer. It is never modified by the FEC. Bits [10–0] are used by the DMA engine, bits[15–11] are ignored.

Table 32-38. Transmit Buffer Descriptor Field Definitions (continued)

Word	Location	Field Name	Description
Offset + 4	Bits [31:16]	A[31:16]	Tx data buffer pointer, bits [31–16][1]
Offset + 4	Bits [15:0]	A[15:0]	Tx data buffer pointer, bits [15–0].

[1] The transmit buffer pointer, which contains the address of the associated data buffer, must always be divisible by 16. The buffer must reside in memory external to the FEC. This value is never modified by the Ethernet controller.

NOTE

Once the software driver has set up the buffers for a frame, it should set up the corresponding BDs. The last step in setting up the BDs for a transmit frame should be to set the R bit in the first BD for the frame. The driver should follow that with a write to TDAR which will trigger the FEC to poll the next BD in the ring.

Chapter 33
Fast Infrared Interface (FIRI)

The Fast Infrared Interface module (FIRI) is capable of establishing any of the following links:

- 0.576 Mbit/s
- 1.152 Mbit/s
- 4 Mbit/s half duplex (using a LED and IR detector)

The FIRI supports these links:

- 0.576 Mbit/s
- 1.152 Mbit/s Medium InfraRed (MIR) physical layer protocol
- 4 Mbit/s Fast InfraRed (FIR) physical layer protocol (defined by IrDA, version 1.4)

FIRI interface signals are multiplexed with UART counterpart signals using a GPIO configuration for a complete infrared interface that supports SIR, MIR, and FIR modes, as illustrated in Figure 33-1. In addition, the Serial InfraRed (SIR) protocol, which supports data rate 115.2 kbps or lower, is implemented in the UART module.

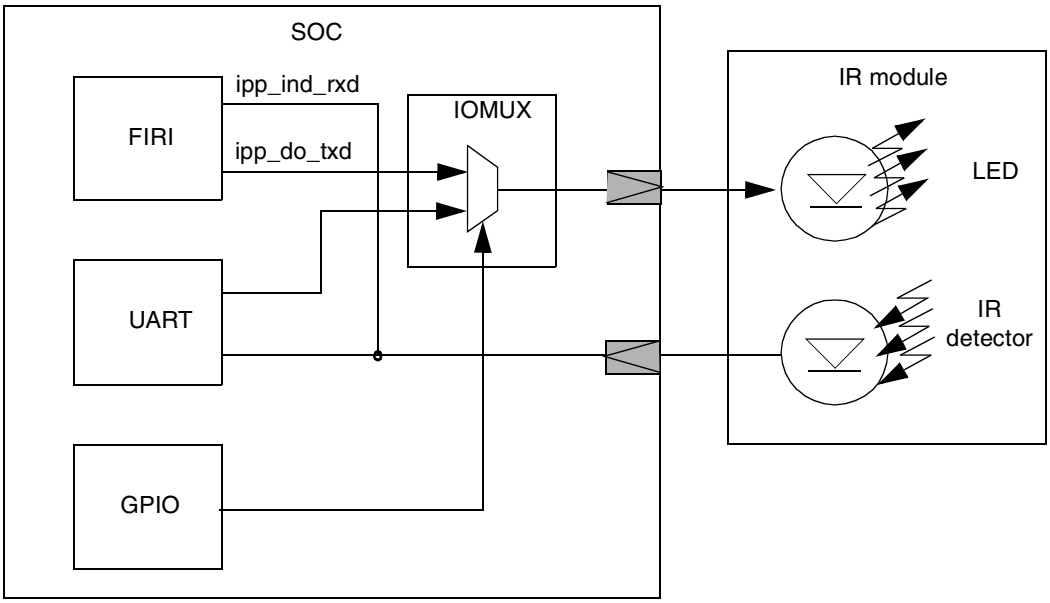


Figure 33-1. FIRI Integration Diagram

33.1 Overview

See Figure 33-2 for an illustration of the hardware components that comprise the FIRI controller.

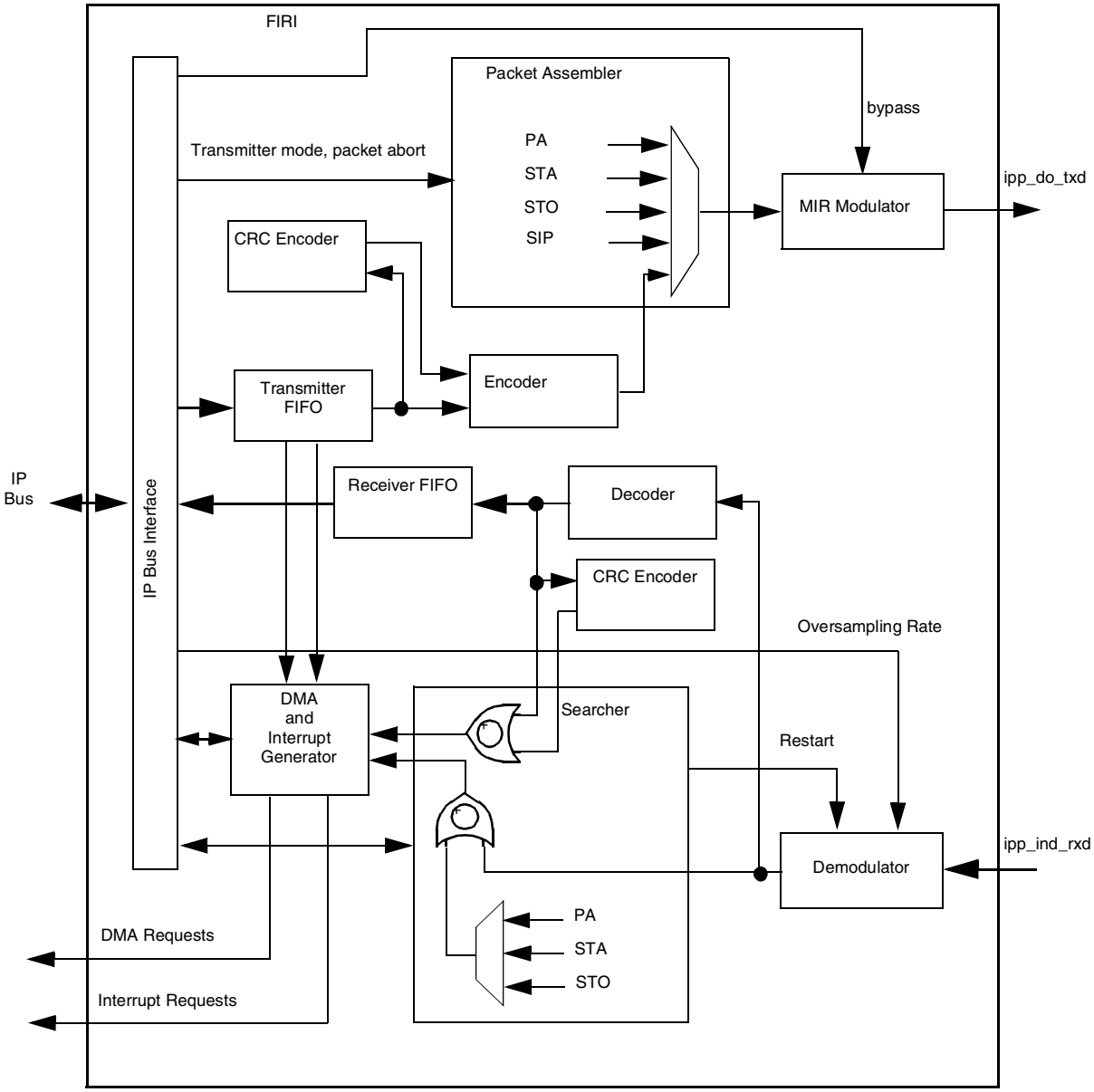


Figure 33-2. FIRI Block Diagram

The FIRI is divided into these four major functional components:

- Transmitter
- Receiver
- FIFO
- IP interface

Each of the four components that make up the FIRI is further made up of individual blocks. The transmitter consists of these blocks:

- Packet Assembler
- CRC Encoder
- MIR Modulator

The receiver consists of these blocks:

- Searcher
- CRC Encoder
- Decoder
- Demodulator

33.1.1 Overview of IrDA Medium InfraRed and Fast InfraRed Standards

The FIRI supports:

- MIR (0.576 Mbit/s, 1.152 Mbit/s)
- FIR (4 Mbit/s) physical layer protocols

This subsection provides a brief overview of the MIR and FIR standards.

33.1.1.1 MIR Packet Structure

The MIR packet format follows the standard high-level data link control (HDLC) format, except that it requires two beginning flags.

The MIR packet consists of:

- Two beginning flags (STA)
- An address
- Control fields
- Data fields
- A frame check sequence (CRC) field
- A minimum of one ending flag (STO)

See Table 33-1 for an illustration of the MIR packet structure, where STA (start flag) and STO (stop flag) are equal, predefined sequences (the left symbol is transmitted/received first).

Table 33-1. MIR Packet Structure

STA	STA	Address	Control & Data	CRC	STO

The fields in Table 33-1 are defined as follows:

- STA, STO

  The MIR links use the same physical layer flag, b’0111,1110, for both STA and STO.
- 8-bit Address Field

- 8-bit Control Field plus up to 2045 bytes in the information field
- CRC field

  The MIR links use a 16-bit CRC-CCITT to check received frames for errors. The CRC is computed from the ADDR and Data fields.

The address, control, data, and CRC fields are not transmitted in original form. They are first converted according to the MIR standards described in the next sections.

33.1.1.2 FIR Packet Structure

See Table 33-2 for an illustration of the 4 Mbit/s FIR data packet format. The chip patterns and symbols for PA, STA, CRC field, and STO are listed in Table 33-2, where PA (preamble), STA, and STO are a predefined sequence (left symbol transmitted/received first).

Table 33-2. FIR Packet Structure

PA	STA	Address & Control & Data	CRC32	STO

The fields in Table 33-2 are defined as follows:

- PA—The preamble field is used by the receiver to establish phase lock. The preamble field consists of exactly sixteen repeated transmissions of the following stream of symbols:

  b’1000,0000,1010,1000
- STA—The STA consists of exactly one transmission of the following stream of symbols:

  b’0000,1100,0000,1100,0110,0000,0110,0000
- STO—The STO consists of exactly one transmission of a stream of symbols:

  b’0000,1100,0000,1100,0000,0110,0000,0110
- ACD—The payload data is encoded as described in the 4 PPM encoding in Table 33-2. The encoded symbols reside in the ACD field and can be up to 2048 bytes long.
- CRC32—The CRC field consists of the 4 PPM encoded data, resulting from the IEEE 802 CRC32 algorithm for cyclic redundancy check as applied to the payload data contained in the packet.

33.1.1.3 MIR CRC

The CRC field is 16 bits long and generated according to the CRC-CCITT algorithm. The CRC polynomial is defined as follows:

Eqn. 33-1

$$CRC(x) = x^{16} + x^{12} + x^5 + 1$$

The CRC(x) value is inverted prior to transmission.

33.1.1.4 FIR CRC

The CRC32 field is 32 bits long and generated according to IEEE 802 CRC32 algorithm. The CRC32 polynomial is defined as follows:

Eqn. 33-2

$$CRC32(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

The CRC(x) value is inverted prior to transmission.

33.1.1.5 MIR Modulation

For MIR data rates, the NZR modulation scheme is used. A “0” is represented by a light pulse. The optical pulse duration is nominally 1/4 of a bit duration. The LED is off when a “1” is transmitted.

Table 33-3. MIR Modulation

Data Bit	Data Symbol (Address, Control, and Data)
0	1000
1	0000

NOTE

Tolerance of bit rate according to the standard must not exceed ±0.1%.

33.1.1.6 FIR Modulation

For 4.0 Mbit/s, the modulation scheme is 4 PPM. In the modulation scheme, a pair of bits is taken together and called a data symbol. A data symbol is divided into four chips, only one of which contains an optical pulse. The nominal pulse duration is 125 ns. A “1” is represented by a light pulse.

Table 33-4. 4 PPM Mapping

Data Bit Pair	4 PPM Data Symbol
00	1000
01	0100
10	0010
11	0001

Verify that the tolerance of chip rate according to the standard does not exceed ±0.01%. The pulse width tolerance of an electrical signal driven by an IR detector (photodiode) can be greater than the tolerance of the optical signal defined by IrDA and can depend on the LED and IR devices used with the FIRI.

33.1.2 Features

The FIRI includes the following distinctive features:

- 0.576 Mbit/s, MIR protocol
- 1.152 Mbit/s, MIR protocol
- 4 Mbit/s, 4 PPM, FIR protocol
- Device Destination Detection Hardware Support
- Interrupt generation
- DMA capability
- SIP generation for collision avoidance

33.1.3 Modes of Operation

The FIRI supports the following modes:

- Hardware packet assembly:
  - FIR mode
  - 0.576 Mbps MIR mode
  - 1.152 Mbps MIR mode
  - Serial Infrared Interaction Pulse (SIP) generation
- Hardware packet search:
  - FIR mode
  - 0.576 Mbps MIR mode
  - 1.152 Mbps MIR mode
- Software packet assembling
- Software packet search

33.2 External Signal Description

See Table 33-5 for the list of signals used to control the FIR.

Table 33-5. Signal Properties

Name	Direction	Port	Function	Reset State	Pull up
Signals Connecting To IC IO					
LED and IR Detector Interface					
IPP_DO_TXD	Output	ipp_do_txd	Data transmit signal. Active high.	1	—
IPP_IND_RXD	Input	ipp_ind_rxd	Data receive signal. Active high.	NA	At top level IO ring, if necessary

33.2.1 Detailed Signal Descriptions

The following list describes the data transmit signal and the data receive signal.

- IPP_DO_TXD (Data Transmit Signal)
  - Output signal from the Packet Assembler module
  - Must be connected to an LED through the IOMUX, as shown in Figure 33-1.
  - A DC-level translator off-chip can be added, if necessary.
- IPP_IND_RXD (Data Receive Signal)
  - Driven by the IR detector
  - Input to the Demodulator module
  - A DC-level translator off-chip can be added, if necessary.

33.3 Memory Map and Register Definition

The FIRI includes six 32-bit registers, a transmit FIFO, and a receive FIFO. Section 33.3.3, Register Descriptions," provides detailed descriptions for all of the FIRI registers.

33.3.1 FIRI Memory Map

Table 33-6 shows the FIRI memory map.

Table 33-6. FIRI Memory Map

Address	Register	Reset Value	Access	Section/Page
0xBASE+0x000 (FIRITCR)	FIRI Transmit Control Register	0x0000_0000	R/W	33.3.3.1/33-9
0xBASE+0x004 (FIRITCTR)	FIRI Transmit Count Register	0x0000_0000	R/W	33.3.3.2/33-11
0xBASE+0x008 (FIRIRCR)	FIRI Receive Control Register	0x0000_0000	R/W	33.3.3.3/33-12
0xBASE+0x00C (FIRITSR)	FIRI Transmit Status Register	0x0000_0000	R/Write one to clear	33.3.3.4/33-14
0xBASE+0x010 (FIRIRSR)	FIRI Receive Status Register	0x0000_0000	R/Write one to clear	33.3.3.5/33-15
0xBASE+0x014	Transmitter FIFO	—	W	33.4.2/33-18
0xBASE+0x018	Receiver FIFO	—	R	33.4.4/33-19
0xBASE+0x01C (FIRICR)	FIRI Control Register	0x0000_0000	R/W	33.3.3.6/33-16

33.3.2 Register Summary

Figure 33-3 shows the key to the register fields and Table 33-7 shows the register figure conventions.

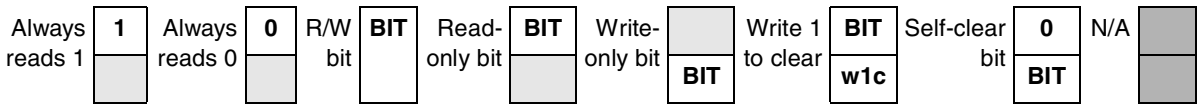


Figure 33-3. Key to Register Fields

Table 33-7. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 33-8 shows the FIRI register summary.

Table 33-8. FIRI Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (FIRITCR)	R	0	0	0	0	0	0	0	HAG	TPA							
	W																
	R	0	SRF		TDT			TCIE	TPEIE	TFUIE	PCF	PC		TPP	TM		TE
	W												SIP				
0xBASE+0x004 (FIRITCTR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	TPL										
	W																
0xBASE+0x008 (FIRIRCR)	R	0	0	0	0	0	0	RAM		RA							
	W																
	R	0	0	0	0	RPEDE	RDT			RPA	RPEIE	PAIE	RFOIE	RPP	RM		RE
	W																
0xBASE+0x00C (FIRITSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	TFP								0	0	0	0	TC	SIPE	TPE	TFU
	W													w1c	w1c	w1c	w1c

Table 33-8. FIRI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x010 (FIRIRSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	RFP								0	0	PAS	RPE	RFO	BAM	CRCE	DDE
	W												w1c	w1c	w1c	w1c	w1c
0xBASE+0x014 Transmitter FIFO	R																
	W	Transmitter FIFO															
	R																
	W	Transmitter FIFO															
0xBASE+0x018 Receiver FIFO	R	Receiver FIFO															
	W																
	R	Receiver FIFO															
	W																
0xBASE+0x01C (FIRICR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	BL							0	OSF			
	W																

33.3.3 Register Descriptions

This section contains the detailed register descriptions for the FIRI registers.

33.3.3.1 FIRI Transmitter Control Register (FIRITCR)

The Transmitter Control Register is a 32-bit, read-write register that controls transmitter operation. Figure 33-4 shows the register; Table 33-9 shows its field descriptions.

0xBASE+0x000 (FIRITCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	HAG	TPA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		SRF		TDT			TCIE	TPEIE	TFUIE	PCF	PC		TPP	TM		TE
W												SIP				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-4. FIRI Transmitter Control Register

Table 33-9. FIRI Transmitter Control Register Field Descriptions

Field	Description
31–25	Reserved
24 HAG	Hardware Address Generator. When this bit is set, the content of the TPA bits is transmitted as a packet address. When the bit is cleared, the packet address is read from TX FIFO. 0 Read packet address from TX FIFO 1 Use TPA bits as packet address
23–16 TPA	Transmit Packet Address. This field contains the 8-bit Transmit Packet Address. If the HAG bit is cleared, the TPA bits have no effect.
15	Reserved
14–13 SRF	Start Field Repeat Factor. This field contains the number of PA or STA fields transmitted in the beginning of the packet for FIR and MIR mode, respectively. In FIR mode, only the 00 value should be used. 00 16 PA or 2 STA fields is transmitted. 01 32 PA or 4 STA fields is transmitted. 10 64 PA or 8 STA fields is transmitted. 11 128 PA or 16 STA fields is transmitted.
12–10 TDT	This field controls the TX FIFO depth that triggers a DMA request. Refer to Section 33.4.2, Transmitter FIFO," for details. **Note:** To avoid TX FIFO overflow, verify that the sum of TDT level and burst length (set by BL bits in the FIRICR register) does not exceed TX FIFO size, which is 128 bytes. 0 DMA request generated when TX FIFO is empty 1 DMA request generated when TX FIFO contain 16 bytes of data 2 DMA request generated when TX FIFO contain 32 bytes of data 3 DMA request generated when TX FIFO contain 48 bytes of data ... ... 7 DMA request generated when TX FIFO contains 112 bytes of data
9 TCIE	Transmit Complete Interrupt Enable. This bit enables the TC status bits of the FIRITSR register to generate a Transmit Complete Interrupt. 0 Transmit Complete Interrupt is not triggered by TC bit. 1 Transmit Complete Interrupt is triggered by TC bit.
8 TPEIE	Transmitter Packet End Interrupt Enable. This bit enables the TPE and SIPE status bits of the FIRITSR register to generate a Packet End Interrupt. 0 PEI interrupt is not triggered by TPE or SIPE bits. 1 PEI interrupt is triggered by TPE or SIPE bits.
7 TFUIE	Transmitter FIFO Underrun Interrupt Enable. This bit enables the TFU status bit of the FIRITSR register to generate a TFU interrupt. 0 TFU interrupt disabled 1 TFU interrupt enabled
6 PCF	Packet Complete by FIFO. This bit determines how a packet is completed if a TX FIFO underrun event occurs. Do not write software intentionally to cause underrun events. However, if it is done due to erroneous conditions, the value of this PC bit selects between two recovery modes. Set the PCF based on system and upper layer IrDA protocol requirements. 0 Send CRC and STO fields 1 Send packet abort symbol

Table 33-9. FIRI Transmitter Control Register Field Descriptions (continued)

Field	Description
5 PC	Packet Complete. This bit determines how a packet is completed when the TE bit is cleared or when the SIP bit is set in the middle of the transfer. Do not write software intentionally to clear the TE bit or to set the SIP bit in the middle of the transfer. However, if it is done due to erroneous conditions, the value of this PC bit selects between two recovery modes. Set PC based on system and upper layer IrDA protocols requirements. 0 Send CRC and STO fields 1 Send packet abort symbol
4 SIP	Transmit Enable of SIP. Writing “1” to this bit produces a “Serial InfraRed Interaction Pulse” transmission. Writing a “0” to this bit is ignored. This bit is always read as “0”. If this bit is set while in the middle of the transfer, the packet will be completed according to the setting of the PC bit.
3 TPP	Transmitter Pulse Polarity bit. 0 Transmitted pulse is not inverted. 1 Transmitted pulse is inverted.
2–1 TM	Transmitter Mode. This bits controls the transmission mode. 00 4 Mbps FIR Mode 01 0.576 Mbps MIR Mode 10 1.152 Mbps MIR Mode 11 Software Packet Assembling
0 TE	Transmitter Enable. This bit controls the FIRI transmitter. If this bit is cleared in the middle of the transfer, the packet will be completed according to the setting of the PC bit. When the packet is completed, the transmitter clocks are then gated OFF. 0 Transmitter Disabled 1 Transmitter Enabled

33.3.3.2 FIRI Transmitter Count Register (FIRITCTR)

The Transmitter Count Register is 32-bit, read-write register that controls the packet size. Figure 33-5 shows the register; Table 33-10 shows its field descriptions.

0xBASE+0x004 (FIRITCTR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R						TPL										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-5. FIRI Transmitter Count Register

Table 33-10. FIRI Transmitter Count Register Field Descriptions

Field	Description
31–11	Reserved
10–0 TPL	Transmit Packet Length bits. The length of the address, control, and data fields (see Table 33-1 and Table 33-2). 0 Send 1 byte 1 Send 2 bytes ... ... 2047 Send 2048 bytes

33.3.3.3 FIRI Receiver Control Register (FIRIRCR)

The Receiver Control Register is 32-bit, read-write register that controls receiver operation. Figure 33-6 shows the register; Table 33-11 shows its field descriptions.

0xBASE+0x008 (FIRIRCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	RAM		RA							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	RPEDE	RDT			RPA	RPEIE	PAIE	RFOIE	RPP	RM		RE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-6. FIRI Receiver Control Register

Table 33-11. FIRI Receiver Control Register Field Descriptions

Field	Description
31–26	Reserved
25–24 RAM	Address Match bit. The value of this bit can be changed when the RE bit is cleared. 00 Does not match address 01 Match packet address to RA bits 10 Match packet address to Broadcast address 11 Match packet address to RA bits and to broadcast address
23–16 RA	Receiver Address. Determines Receiver Packet Address. If the RAM bit value is 00 or 10, the RA bit has no effect. The value of this bit can be changed when the RE bit is cleared.
15–12	Reserved

Table 33-11. FIRI Receiver Control Register Field Descriptions (continued)

Field	Description
11 RPEDE	Receiver Packet End DMA Request Enable bit; Enable DMA request generation at the end of the packet. If this bit is set, verify that the BL value in the FIRICR register is not greater than the sum of the address, control, data, and CRC field length. 0 DMA request is not affected by the end of packet. 1 DMA request is generated at the end of packet.
10–8 RDT	Receiver DMA Request Trigger level. Sets minimum Rx FIFO depth, which triggers a DMA request. For more details, refer to Section 33.4.4, Receiver FIFO." **Note:** To avoid Rx FIFO underflow, set the RDT level to greater than or equal value of the burst length (set by the BL bit in the FIRICR register). 0 Reserved 1 DMA request generated when Rx FIFO contains 16 bytes of data 2 DMA request generated when Rx FIFO contains 32 bytes of data 3 DMA request generated when Rx FIFO contains 48 bytes of data ... 7 DMA request generated when Rx FIFO contains 112 bytes of data
7 RPA	Receiver Packet Abort bit. Determines behavior of the Rx FIFO upon detection of an illegal symbol. When an illegal symbol is detected, the DDE or CRCE bit in the FIRIRSR register is set. If the RPA bit is set, the Rx FIFO pointers are cleared and the receiver starts to search for the PA or STA fields for FIR and MIR mode, respectively. If RPA is cleared, the receiver continues to write to the Rx FIFO. 0 Does not clear the Rx FIFO upon detection of an illegal symbol 1 Clears the Rx FIFO upon detection of illegal symbol
6 RPEIE	Receiver Packet End Interrupt Enable bit. Enables the RPE status bit in the FIRIRSR register to assert the Packet End Interrupt. 0 PEI interrupt is not triggered by RPE bit. 1 PEI interrupt is triggered by RPE bit.
5 PAIE	Packet Abort Interrupt Enable bit. Enables the DDE, CRCE status bits in the FIRIRSR register to cause a PAI interrupt. 0 PAI interrupt is disabled. 1 PAI interrupt is enabled.
4 RFOIE	Receiver FIFO Overrun Interrupt Enable bit. Enables the RFO status bit in the FIRIRSR register to cause a RFOI interrupt. 0 RFOI interrupt is disabled. 1 RFOI interrupt is enabled.
3 RPP	Receiver Pulse Polarity bit. 0 Receiver signal is not inverted. 1 Received signal is inverted.
2–1 RM	Receiver Mode bits. 00 FIR Mode 01 0.576 Mbps MIR Mode 10 1.152 Mbps MIR Mode 11 Software Packet Assembling
0 RE	Receiver enable bit. When this bit is cleared, the receiver clocks are gated OFF. 0 Receiver is disabled. 1 Receiver is enabled.

33.3.3.4 FIRI Transmit Status Register (FIRITSR)

The Transmit Status Register is 32-bit register that reflects the status of the transmitter and the TX FIFO. The FIRITSR contains read-only and write-one-to-clear bits. Figure 33-7 shows the register; Table 33-12 shows its field descriptions.

0xBASE+0x00C (FIRITSR) Access: User read/write one to clear

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TFP								0	0	0	0	TC	SIPE	TPE	TFU
W													w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-7. FIRI Transmit Status Register

Table 33-12. FIRI Transmit Status Register Descriptions

Field	Description
31–16	Reserved
15–8 TFP	Transmitter FIFO Pointer bits. The value of the TFP bits represent the number of available bytes in TX FIFO (read-only bits).
7–4	Reserved
3 TC	Transmit Complete bit. Indicates that the transmission is completed (including the CRC and STO fields), and the transmitter FIFO is empty. This bit is cleared by writing a “1”. 0 Transmitting is not completed. 1 Transmitting is completed.
2 SIPE	SIP End bit. Indicates that “Serial InfraRed Interaction Pulse” has been transmitted. This bit is cleared by writing a “1”. 0 SIP transmission is not completed. 1 SIP had been transmitted.
1 TPE	Transmitter Packet End bit. Indicates that the TPS bytes of data (address, control, and data fields) have been transmitted. This bit is cleared by writing a “1”. 0 Transmissions of address, control, and data fields are not completed. 1 Address, control, and data fields had been transmitted.
0 TXU	Transmitter FIFO Underrun bit. Indicates the occurrence of a TX FIFO underrun. A TX FIFO underrun occurred if the transmitter attempted to read from the TX FIFO when the TX FIFO is empty. This bit is cleared by writing a “1”. 0 No transmitter FIFO underrun 1 Transmitter FIFO is underrun

33.3.3.5 FIRI Receive Status Register (FIRIRSR)

The Receive Status Register is 32-bit register that reflects the status of the receiver and the FIFO. It contains read-only and write-one-to-clear bits. Figure 33-8 shows the register; Table 33-13 shows its field descriptions.

0xBASE+0x010 (FIRIRSR) Access: User read/write one to clear

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RFP								0	0	PAS	RPE	RFO	BAM	CRCE	DDE
W												w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-8. FIRI Receive Status Register

Table 33-13. FIRI Receive Status Register Field Descriptions

Field	Description
31–16	Reserved
15–8 RFP	Receiver FIFO Pointer bits. The value of the RFP bits represent the number of available bytes in the Rx FIFO (read-only bits).
7–6	Reserved
5 PAS	Preamble Search bit. Indicates that the receiver is searching for PA+STA, or STA fields for FIR and MIR modes, respectively (read-only bit). 0 Receiver does not search for the PA or STA field. 1 Receiver searches for the PA or STA field.
4 RPE	Receiver Packet End bit. Indicates that the STO field or packet abort symbol (b'0000,000 and b'0000,0000 for MIR and FIR, respectively) is detected. This bit is cleared by writing a "1". 0 STO was not detected. 1 STO field is detected.
3 RFO	Receiver FIFO Overrun bit. Indicates that the receiver attempted to write to the Rx FIFO when the Rx FIFO is full. This bit is cleared by writing a "1". 0 Receiver FIFO is not overrun. 1 Receiver FIFO is overrun.
2 BAM	Broadcast Address Match bit. Indicates that the address field of the packet matches the Broadcast Address 0xFF. This bit is cleared by writing a "1". 0 No Broadcast 1 Broadcast packet

Table 33-13. FIRI Receive Status Register Field Descriptions (continued)

Field	Description
1 CRCE	CRC Error bit. Indicates that the CRC check failed. This bit is cleared by writing a “1”. 0 No CRC check failure 1 CRC check failure
0 DDE	Address, Control, or Data Field Error bit. Indicates that an illegal symbol has been detected in the address, control, data, or CRC32/CRC fields. This bit is cleared by writing a “1”. 0 No illegal symbols in address, control, data, or CRC field 1 Illegal symbol in address, control, data, or CRC field

33.3.3.6 FIRI Control Register (FIRICR)

The Control Register is a 32-bit, read-write register that is shared by receiver and transmitter. Figure 33-9 shows the register; Table 33-14 shows its field descriptions.

0xBASE+0x01C (FIRICR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	BL							0	OSF			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 33-9. FIRI Control Register

Table 33-14. FIRI Control Register Field Descriptions

Field	Description
31–12	Reserved
11–5 BL	Burst Length. Used to prevent an overrun of the TX FIFO and an underrun of the Rx FIFO. The DMA request of the transmitter deasserts when the number of vacant bytes in the TX FIFO is less than the burst length settings specified by the BL bits. The DMA request of the receiver deasserts when the number of available bytes in the Rx FIFO is less than the burst length settings specified by the BL bits. 1 1 byte 2 2 bytes 3 3 bytes ... 127 127 bytes 0 128 bytes

Table 33-14. FIRI Control Register Field Descriptions (continued)

Field	Description
4	Reserved
3–0 OSF	Over Sampling Factor. This field controls the oversampling factor of the “chip” on the IPP_IND_RXD signal if the RE bit in the FIRIRCR register is set, or the prescaling factor of the IPG_CLK_FIRI_BAUD signal if the TE bit in the FIRITCR register is set. **Note:** The “chip” rate is 4 and 2 times greater than the bit rate for MIR and FIR, respectively. 0 Does not oversample 1 Oversamples by 2 2 Oversamples by 3 ... ... 15 Oversamples by 16

To enable proper receive operations, the period of the divided clock must be four or more times shorter than the pulse width of the IPP_IND_RXD signal, and one or more times shorter than the gap between two consecutive pulses.

NOTE

The pulse width of the IPP_IND_RXD signal is dependent upon the characteristics of the LED and the Photo Diode used with the FIRI, and is different from the pulse width defined by IrDA for the optical signal. The value of the OSF bits must be adjusted accordingly. A greater OSF value leads to a better SNR on the expense of an increase in the power consumption.

33.4 Functional Description

This section details the FIRI functional description.

33.4.1 Transmitter Overview

The transmitter has three modes of operation:

- MIR
- FIR
- Software Packet Assembly

33.4.1.1 MIR Mode

In MIR mode, the following sequence occurs:

1. The Packet Assembler block sends the STA sequences (two or more times) to the MIR Modulator block.
2. The Encoder block accesses the TX FIFO, aligns the data, and sends the ACD (Address, Control, and Data) fields to the Packet Assembler block.

3. The CRC field calculated by CRC block follows the ACD field; after which, the STO bits are sent to the Modulator.

33.4.1.2 FIR Mode

In FIR mode, the following sequence occurs:

1. The Packet Assembler block sends the predefined PA (16 or more times) and STA sequences to the LED.
2. The Encoder block accesses the TX FIFO, encodes the data (4 PPM), and sends the ACD (Address, Control and Data) to the Packet Assembler block.
3. The CRC32 field calculated by CRC block follows the ACD field; after which, the STO bits are sent to the Modulator.

NOTE

The MIR Modulator block is not operable in FIR mode.

If the DMA request is not served during MIR or FIR modes, and there is no valid data to transmit, the Assembler Module terminates the packet using the packet-abort symbol (with CRC/CRC32 and STO fields [see the PCF bit description]), and a TFUI interrupt is generated. The packet can also be aborted by the packet length counter or by the software (see PC bit description).

33.4.1.3 Serial Infrared Interaction Pulse

The transmitter must send the “Serial Infrared Interaction Pulse” (SIP) at 500 ms to guarantee that low speed IR devices will not interfere with its pulses. The pulse must be 8.7 µs wide; and positive phase must be 1.6 µs wide.

33.4.1.4 Software Packet Assembly Mode

In Software Packet Assembly mode, the entire packet is read from the TX FIFO, including the PA, STA, and STO bits. The MIR modulator and Encoder blocks are disabled. This mode is used for debug purposes and can be used for testing compatibility with future IrDA protocols, if appropriate.

33.4.2 Transmitter FIFO

The transmitter FIFO is a 128-byte FIFO used for holding transmit data. The TX FIFO gets filled by the accesses to the Transmitter FIFO address (by core or DMA) and is emptied by the transmitter. When the DMA is used, a pre-programmed FIFO trigger level controls the triggering of the DMA request. When the TX FIFO is emptied below the trigger-level, a DMA request signal is asserted. The DMA request gets deasserted when the number of vacant bytes in the TX FIFO is less than the burst length settings specified by the BL bits in the FIRICR register. If the DMA request was not serviced and, as a result, there are no more data in the TX FIFO, the TFU bit in the FIRITSR register sets, and the $\overline{\text{IPI\_INT\_FIFO}}$ interrupt signal asserts depending on TFOIE bit in the FIRITCR register. During the transmit operation, the TFP bit value in the FIRITSR register reflects the amount of available bytes in the TX FIFO.

33.4.3 Receiver Overview

The receiver has three modes of operation:

- MIR
- FIR
- Software Packet Disassembly Modes

In MIR and FIR modes, each incoming pulse is oversampled by the programmable factor in the Demodulator block. Next, the Demodulator block detects the edges of the pulses and synchronizes to the phase of the transmitted pulse. Phase correction is performed until the end of the packet because the bit rate tolerance accumulates to very large values for long packets.

33.4.3.1 MIR Mode

In MIR mode, the Searcher block searches for the STA field. If the sequence is matched, the data stream following the STA field is sent to the Decoder block. The Decoder block searches for five consecutive “1”s and skips next the “zero” bit inserted by the transceiver for phase synchronization. The data is then written to the Rx FIFO. In parallel, the Searcher Module searches for the STO field. When the STO field is detected, a PEI interrupt is generated together with a DMA request, and the corresponding flag is set in the status register. The Searcher module continuously searches for illegal symbols (seven or more consecutive “1”s). If an illegal symbol is detected, the PAI interrupt is generated and the corresponding flag is set in the status register.

33.4.3.2 FIR Mode

In FIR mode, the Searcher Module searches for the PA field and then for the STA field. If found, the “chip” stream is converted to a data stream by the Decoder block and then it is written to the Rx FIFO. In a parallel operation, the Searcher Module searches for the STO field. When the STO field is detected, a PEI interrupt can be generated together with a DMA request. While receiving, the Searcher module continuously searches the PA, STA, ACD, CRC32, and STO fields for illegal symbols. In addition, the Searcher module looks for a packet abort symbol. On detection of an illegal symbol, a PAI interrupt is generated and the corresponding flag is set in the status register.

In MIR and FIR modes, the “Address” bits can be compared to a predefined value or/and to the broadcast value 0xFF (see the description of RAM bits). If the CRC field value does not match an expected value calculated by the CRC Encoder block, a PAI interrupt is generated. The CRC field sends the Rx FIFO, irrespective of a comparison result.

33.4.3.3 Software Packet Disassembly Mode

In Software Packet Disassembly mode, the entire packet is sent to the Rx FIFO, even if illegal symbols have been detected.

33.4.4 Receiver FIFO

The receiver FIFO is a 128-byte FIFO, used for holding received data. Rx FIFO gets filled by incoming data and emptied by excess Rx FIFO (by the core or DMA). When DMA is used, a pre-programmed FIFO

trigger level controls the triggering of the DMA request. When Rx FIFO is filled beyond the trigger level, a DMA request signal is asserted. The DMA request gets deasserted when the number of available bytes in Rx FIFO is less than the burst length settings specified by the BL bits in the FIRICR register. If the DMA request was not serviced and, as a result, there are no vacant bytes in the Rx FIFO for receive data, the RFO bit in the FIRIRSR register is set and the IPI_INT_FIFO interrupt signal is asserted depending on RFOIE bit in the FIRIRCR register. During a receive operation, the RFP bits' value in the FIRIRSR register reflects the amount of available bytes in the Rx FIFO.

33.5 Initialization/Application Information

This section provides the following two examples of FIRI programming: a transmitter programming scenario and a receiver programming scenario.

33.5.1 Transmitter Programming Scenario

The purpose of this scenario is to transmit 1024 bytes in FIR mode using an infrared link. Verify that the packet length is 512 bytes, and then use the following steps.

1. Configure the clock controller. For this example, the frequency of the IPG_CLK_FIRI_BAUD clock is 48 MHz.
2. Configure the DMA module for DMA-service FIRI transactions. A DMA transaction is requested by the negative edge of DMA_REQ[1]. The DMA access size is 4 bytes. For this example, the burst length is 4 accesses.
3. Configure the FIRI to work with a 32-byte burst (FIRICR.BL bits).

NOTE

Observe the burst length of the DMA and the burst length selected by FIRICR. The BL bits are not necessarily equal.

4. Set the FIRI Oversampling Factor (FIRICR.OSF bits) to 6 to achieve a 4-Mbit rate with the IPG_CLK_FIRI_BAUD clock.
5. Configure the FIRI to transmit 512-byte long packets (FIRITCTR.TPL bits).
6. Set the DMA trigger level to 16 (FIRITCR.TDT bits). A DMA request is generated when there are only 16 bytes remaining in the TX FIFO. Select FIR mode (FIRITCR.TM bits).
7. Finally, enable the transmitter (FIRITCT.TE bit). The FIRI immediately asserts a DMA request because the TX FIFO is empty. The DMA controller delivers a first burst to the FIRI. The FIRI starts the transmission.

33.5.1.1 Receiver Programming Scenario

Fort the receiver programming scenario, the data is transmitted in FIR mode. An interrupt should be generated at the end of each packet.

1. Configure the clock controller. For this example, the frequency of the IPG_CLK_FIRI_BAUD clock is 48 MHz.

2. Configure the DMA module for the FIRI to DMA transactions. The DMA transaction is requested by the negative edge of $\overline{\text{DMA\_REQ}}$[0]. DMA access size is 4 bytes. For this example, burst length is 4 accesses.
3. Configure the FIRI to work with 16-byte burst (FIRICR.BL bits). Set the FIRI Oversampling Factor (FIRICR.OSF bits) to 6 to achieve a 4-Mbit rate with the IPG_CLK_FIRI_BAUD clock.
4. Set the DMA trigger level to 16 (FIRIRCR.RDT bits). A DMA request is generated when there are 16 bytes in Rx FIFO. Select FIR mode (FIRIRCR.RM bits). Enable Packet End Interrupt.
5. Finally, enable the receiver (FIRIRCR.RE bit). The receiver searches for the PA and STA fields. When a PEI interrupt is generated, verify that the software clears the FIRIRSR.RPE bit.

Chapter 34
General Power Controller (GPC)

34.1 Introduction

GPC is a general power control module. The GPC block includes the following subblocks of advanced power saving techniques:

- Two DVFS subblocks (CPU and peripherals clock/power domains)
- Two DPTC subblocks (CPU and peripherals power domains)
- SRPG subblock for all SRPG-ed modules: CPU, VPU, GPU, IPU, EMI, LPMIX, and SIPMIX

Each of the subblocks has its own IP registers. GPC also includes an arbitration block for DVFS and DPTC voltage/frequency updates.

Figure 34-1 shows the GPC block diagram.

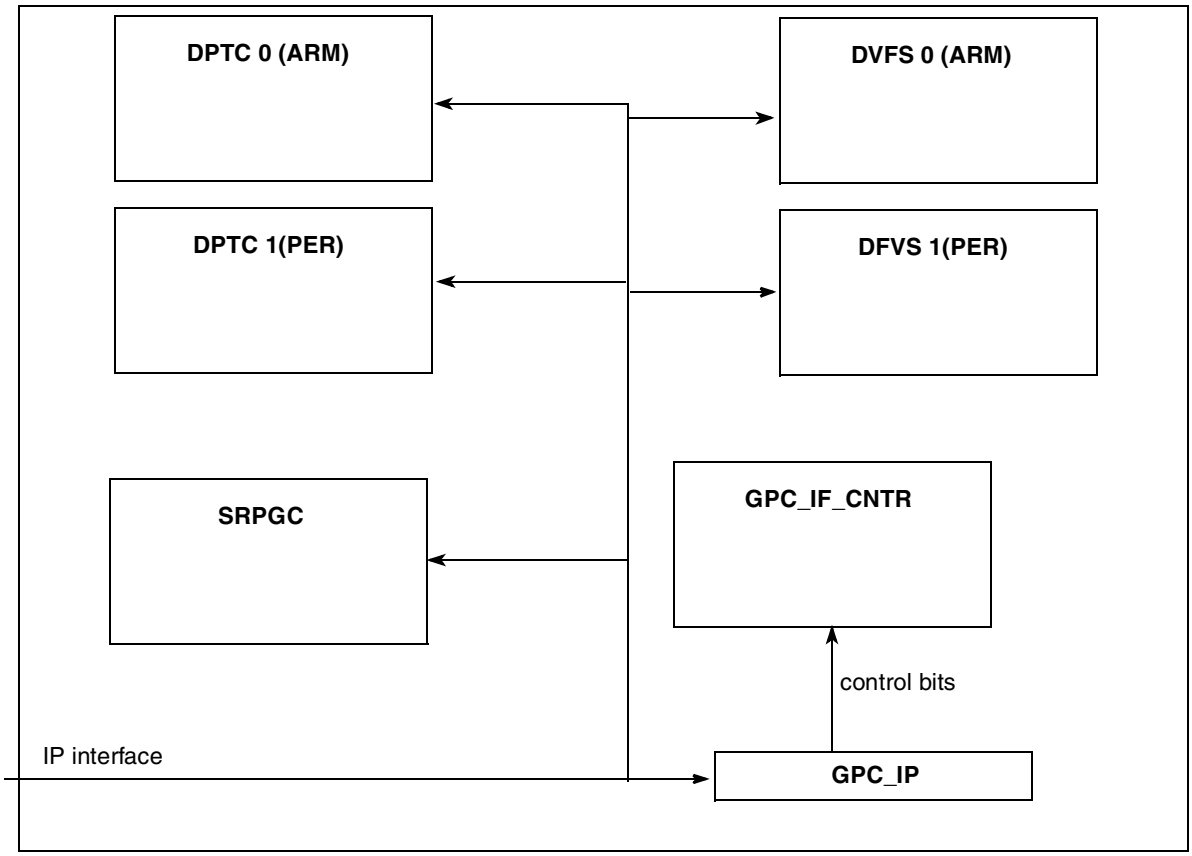


Figure 34-1. GPC Block Diagram

34.1.1 Features

The main features are as follows:

- Simultaneous usage of DVFS and DPTC of each domain.
- Internal counter for bypass of voltage_ready signal
- Possibility of switching between CSPI usage and I2C.

34.2 Memory Map and Register Definition

This section provides a memory map, a key to register conventions, and detailed descriptions of all registers.

34.2.1 Memory Map

Table 34-1 shows the memory map.

Table 34-1. Memory Map [1]

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x000	CNTR: GPC block—Interface control register	R/W	0210_8000	34.2.3/34-5
0xBASE+0x004	PGR: GPC block—Power Gating Register	R/W	0000_0000	34.2.4/34-6
0xBASE+0x008	VCR: GPC block—Voltage Counter Register	R/W	0000_0001	34.2.5/34-8
0xBASE+0x00C	ALL_PU : GPC block—ALL_PU Register	R/W	0000_0700	34.2.6/34-9
0xBASE+0x010	NEON : GPC block NEON register	R/W	0000_0030	34.2.7/34-10
0xBASE+0x080 … 0xBASE+0x094	DPTC_LP registers	R/W	DPTC spec	
0xBASE+0x100 … 0xBASE+0x114	DPTC_GP registers	R/W	DPTC spec	
0xBASE+0x180 … 0xBASE+0x1C0	DVFS CORE registers Base addr = BASE_GPC + 0x0000_0180	R/W	DVFS CORE spec	
0xBASE+0x1C4 … 0xBASE+0x1E0	DVFS PER. registers Base addr = BASE_GPC + 0x0000_01C4	R/W	DVFS PER spec	
0xBASE+0x220 … 0xBASE+0x22C	PGC IPU	R/W	PGC block spec	
0xBASE+0x240 … 0xBASE+0x24C	PGC VPU	R/W	PGC block spec	

Table 34-1. Memory Map (continued)[1]

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x260 … 0xBASE+0x26C	PGC GPU	R/W	PGC block spec	
0xBASE+0x280 … 0xBASE+0x290	SRPGC NEON	R/W	SRPGC block spec	
0xBASE+0x2A0 … 0xBASE+0x2B0	SRPGC TIGER	R/W	SRPGC block spec	
0xBASE+0x2C0 … 0xBASE+0x2CC	EMPGC0 TIGER (CTA8_L1)	R/W	EMPGC block spec	
0xBASE+0x2D0 … 0xBASE+0x2DC	EMPGC1 TIGER (CTA8_L2)	R/W	EMPGC block spec	
0xBASE+0x2E0 … 0xBASE+0x2F0	SRPGC MEGAMIX	R/W	SRPGC block spec	
0xBASE+0x300 … 0xBASE+0x310	SRPGC EMI	R/W	SRPGC block spec	

1 See Chapter 2, “Memory Map,” for the value of base address of GPC.

34.2.2 Register Summary

The conventions in Figure 34-2 and Table 34-2 serve as a key for the register summary and individual register diagrams.

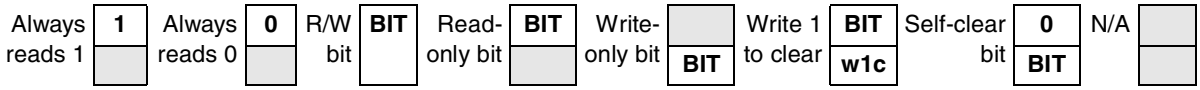


Figure 34-2. Key to Register Fields

Table 34-2 provides a key for register figures and tables and the register summary.

Table 34-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.

Table 34-2. Register Conventions (continued)

Convention	Description
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 34-3 shows the GPC register summary.

Table 34-3. GPC Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNTR 0xBASE+0x000	R	0	0	0	0	0	CSPI	IRQ2M	IRQ2	0	0	GPCIRQM	GPCIRQ	DPTC1CR	DPTC0CR	DVFS1CR	DVFS0CR
	W																
	R	ADU	STRT	FUPD							0	0	0	HTRI			
	W																
PGR 0xBASE+0x004	R		DRCIC		IPCC	IPCO						IPCI					
	W																
	R							CTA8PG				GPUPG		VPUPG		IPUPG	
	W																
VCR 0xBASE+0x008																VINC	VCNTU
		VCNT															

Table 34-3. GPC Register Summary (continued)

Name	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ALL_PU 0xBASE+0x00C																
						VPU SW-STATUS	GPU SW-STATUS	IP-US WSTATUS		VP-UPUR	GP-UPUR	IPU-PUR		VP-UPDR	GP-UPDR	IP-UPDR
NEON 0xBASE+0x010																
											NEONF-SMST				NEONPUR	NEONPDR

34.2.3 CNTR Register Description

CNTR Register

Address 0xBASE+0x000 Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	CSPI	IRQ2M	IRQ2	0	0	GPCIRQM	GPCIRQ	DPTC1CR	DPTC0CR	DVFS1CR	DVFS0CR
W																
Reset						0	1	0			0	1	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADU	STRT	FUPD	0	0	0	0	0	0	0	0	0	HTRI			
W																
Reset	1	0	0										0000			

Figure 34-3.

Describe register functions in tables, as shown in Table 34-4.

Table 34-4. Register Field Descriptions

Field[1]	Description
31-27	reserved
26	CSPI - CSPI or I2C is used (hw trigger will be generated or irq2 will be sent for PMIC) 0 - default - I2C is in use 1 - CSPI is in use
25	IRQ2M - int2 (for I2C) masking 0 - int2 will be sent by GPC FSM if CSPI is ‘0’ (I2C is in use) 1 - default - int2 is masked, but GPC FSM will wait for INT2 bit to be negated by writing ‘1’, if CSPI is ‘0’.

Table 34-4. Register Field Descriptions (continued)

Field[1]	Description
24	IRQ2 - status bit, write 1 to clear.
23-22	Reserved
21	GPCIRQM - GPC interrupt/event masking 1 - interrupt/event is masked 0 - not masked
20	GPCIRQ - GPC will generate ARM IRQ or SDMA event, as a result of DVFS or DPTC change requests 1 - GPC will generate ARM IRQ 0 - GPC will generate SDMA event
19	DPTC1CR - DPTC1 (PER) Change request (bit is read-only) 1 - DPTC1 is requesting for voltage update 0 - DPTC1 has no request
18	DPTC0CR - DPTC0 (ARM) Change request (bit is read-only) 1 - DPTC0 is requesting for voltage update 0 - DPTC0 has no request
17	DVFS1CR - DVFS1 (PER) Change request (bit is read-only) 1 - DVFS1 is requesting for frequency/voltage update 0 - DVFS1 has no request
16	DVFS0CR - DVFS0 (ARM) Change request (bit is read-only) 1 - DVFS0 is requesting for frequency/voltage update 0 - DVFS0 has no request
15	ADU - ARM domain freq/voltage update needed 1 - ARM domain frequency and/or voltage update needed 0 - PER domain frequency and/or voltage update needed
14	STRT - Controller start Controller operation will be started when bit set to “1”. Bit will be set automatically to “0” when frequency / voltage change finished. There is still a possibility to write “0” by s/w. 1 - Controller operation in progress. No IRQ is allowed during voltage update procedure (IRQ will be masked) 0 - Controller operation finished. New freq/voltage change request is available
13	FUPD - frequency update needed 1 - frequency update needed 0 - frequency updated is not needed
12-4	Reserved
3-0	HTRI - Hardware Triggering Register Index Indicates which one of the CSPI h/t register will be written to the PMIC. 0000 - register #0 will be triggered, 0001 - register #1 will be triggered, 0010 - register #3 will be triggered...

[1] This footnote applies to the entire column. Field columns in a register field description table should use TBItem_C paragraph format.

34.2.4 PGR - Register Description

PGR - Power Gating Register

Address 0xBASE+0x004 Access: User read-only

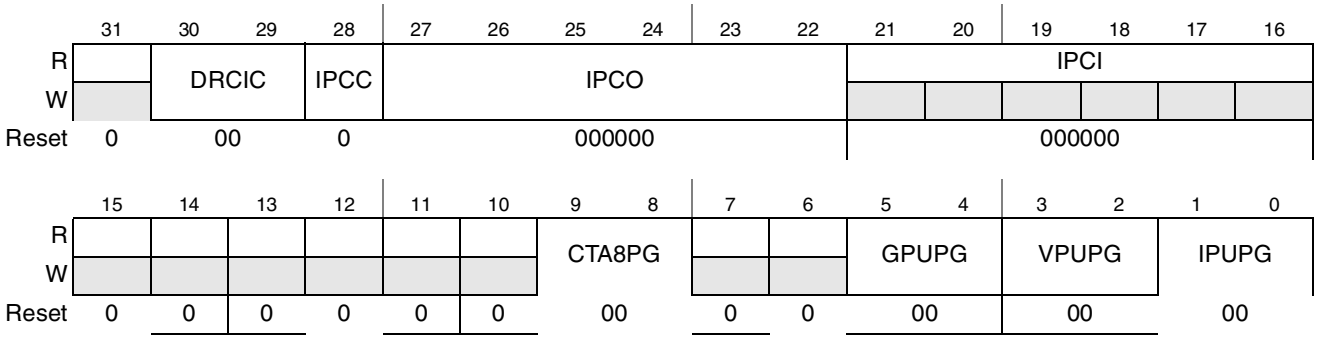


Figure 34-4. 32-Bit Register Format—Two Line

Describe register functions in tables, as shown in Table 34-5.

Table 34-5. Register Field Descriptions

Field	Description
31	Reserved
29-30	DRCIC - DPTC ref cir in mux control 00 - ref_clk_lp_0 01 - ccm_cosr_1_clk_in 10 - ccm_cosr_2_clk_in 11 - io_pvt_clk_in
28	IPCC - IO PTV Control Configuration - output to PVT 0 - use internal oscillator 1 - use external (DPTC) oscillator, value from IPCO
22-27	IPCO - IO PTV Control Out - value of external (DPTC) oscillator for calculating strength of IO pads, provided by SW
16-21	IPCI - IO PTV Control In - value of internal PVT oscillator for calculating strength of IO pads
10-15	Reserved
8-9	CTA8PG - CTA8 (TIGER) Power Gating Power gating according to Table 34-6
6-7	Reserved
4-5	GPU - GPU Power Gating Power gating at Stop Mode (when CCM's "stop" signal asserted)
2-3	VPU - VPU Power Gating Power gating at Stop Mode (when CCM's "stop" signal asserted)
0-1	IPU - IPU Power Gating Power gating at Stop Mode (when CCM's "stop" signal asserted)

Table 34-6. Power Gating bits description

PG bits	Power Gating conditions
00	Power is never Gated
01	Power Gated in Wait Mode
10	Power Gated in Doze Mode
11	Power Gated in Stop Mode

34.2.5 VCR Register Description

VCR Register - Voltage Counter Register

Address 0xBASE+0x008 Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VINC	VCN-TU
W																
Reset															0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	VCNT														
W																
Reset	000_0000_0000_0001															

Figure 34-5. VCR Register Diagram

Describe register functions in tables, as shown in Table 34-7.

Table 34-7. Register Field Descriptions

Field[1]	Description
31-18	Reserved
17	VINC - Voltage increase 1 - Voltage will be increased 0 - Voltage will be decreased
16	VCNTU - Voltage Count Used 1 - VCNT is used 0 - VCNT is not used, “voltage ready” signal of PMIC will be used.
14-0	VCNT - Voltage update Count Counting delay of the voltage update, if PMIC’s “voltage ready” signal is not used Counter is using SYS_CLK (CKIH) clock

[1] This footnote applies to the entire column. Field columns in a register field description table should use TBItem_C paragraph format.

34.2.6 ALL_PU Register Description

ALL_PU Register - Register Control Power for all PUs

Address 0xBASE+0x00C — Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	VPUSWSTATUS	GPUSW-STA-TUS	IP-USW-STATUS	0	VPU-PUR	GPU-PUR	IPU-PUR	0	VP-UPDR	GP-UPDR	IPUP-DR
W																
Reset						1	1	1		0	0	0		0	0	0

Figure 34-6. ALL_PU Register Diagram

Describe register functions in tables, as shown in Table 34-7.

Table 34-8. Register Field Descriptions

Field[1]	Description
31-11	Reserved
10	VPUSWSTATUS - status of latest sw request. Reset value - 1'b1, "power up". 1 - software power up request was asserted for VPU 0 - software power down request was asserted for VPU
9	GPUSWSTATUS - status of latest sw request. Reset value - 1'b1, "power up". 1 - software power up request was asserted for VPU 0 - software power down request was asserted for VPU
8	IPUSWSTATUS - status of latest sw request. Reset value - 1'b1, "power up". 1 - software power up request was asserted for VPU 0 - software power down request was asserted for VPU
7	Reserved
6	VPUPUR - software VPU Power Up Request. VPU PGC power up request is OR between SW and HW signals
5	GPUPUR - software GPU Power Up Request. GPU PGC power up request is OR between SW and HW signals
4	IPUPUR - software IPU Power Up Request. IPU PGC power up request is OR between SW and HW signals
3	Reserved
2	VPUPDR - software VPU Power Down Request. VPU PGC power down request is OR between SW and HW signals
1	GPUPDR - software GPU Power Down Request. GPU PGC power down request is OR between SW and HW signals
0	IPUPDR - software IPU Power Down Request. IPU PGC power down request is OR between SW and HW signals

[1] This footnote applies to the entire column. Field columns in a register field description table should use TBItem_C paragraph format.

34.2.7 NEON Register Description

NEON Register - Register Control Power for NEON

Address 0xBASE+0x010 Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset																

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	NEONF-SMST		0	0	NEO-NPUR	NEO-NPDR
W																
Reset											1	1			0	0

Figure 34-7. NEON Register Diagram

Describe register functions in tables, as shown in Table 34-7.

Table 34-9. Register Field Descriptions

Field[1]	Description
31-6	Reserved
5-4	NEONFSMST - Neon FSM status bits - state of Neon State Machine:g 11 - Power up acknowledge / Normal state 01 - Power down request 00 - Power down acknowledge 10 - Power up request
3-2	Reserved
1	NEONPUR - NEON Power Up Request. Assertion causes Neon PowerDown state machine to change state from PowerDown to PowerUp request. Will clear itself after a single clock.
0	NEONPDR - NEON Power Down Request. Assertion starts Neon PowerDown state machine. Will clear itself after a single clock.

1 This footnote applies to the entire column. Field columns in a register field description table should use TBItem_C paragraph format.

34.3 Functional Description

GPC block includes the following sub-blocks of Advanced Power Saving techniques:

- 2 DVFS sub-blocks (CPU and peripherals clock/power domains)
- 2 DPTC sub-blocks (CPU and peripherals power domains)
- SRPG sub-block for all SRPG-ed modules: CPU, VPU, GPU, IPUv3, EMIv2, LPMIX and SIPMIX

Each of sub-blocks has its own IP registers. As well, GPC includes an arbitration block of DVFS and DPTC voltage/frequency updates.

In the following sections the global operation of the sub-blocks is described. For further details of DPTC, DVFS and SRPG refer appropriate sub-block Block Guide document.

34.3.1 DVFS — Dynamic Voltage & Frequency Scaling

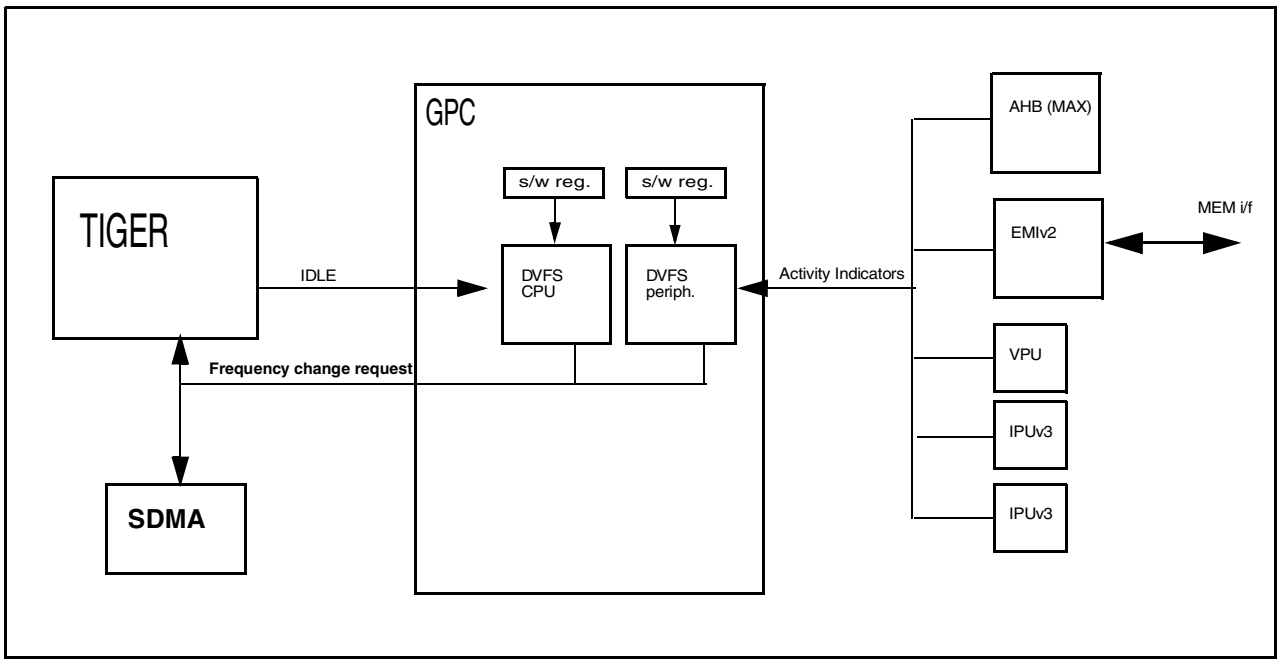
Dynamic Voltage & Frequency Scaling is a well-known technique to reduce power consumption in mobile devices.

In order to improve power saving efficiency, DVFS is applied both on CPU domain and peripherals domain. CPU-domain DVFS uses mainly ARM's IDLE signal for load monitoring, but also uses s/w writable info as secondary inputs. Peripherals-domain DVFS uses hardware accelerators activity signals, bus activity signals and memory activity indicators for load monitoring. Both DVFS load tracking controllers are h/w implemeted, providing: high resolution of load tracking (for higher efficiency), no MIPS requirements from CPU for continuous operation (more power saving, no additional CPU load).

Voltage update requests (to PMIC, via CSPI) may be served in one of the following ways:

- Hardware triggering of CSPI, by GPC
- ARM interrupts serving CSPI

Figure 34-8. DVFS System High Level Diagram



34.3.2 DPTC — Dynamic Process and Temperature Compensation

Dynamic process and temperature compensation is an advanced power saving technique that allows dynamic adjustment of the supply voltage relative to the process corner case and temperature of the chip.

While design of an IC is always done for the worst case process and hot temperature assumptions, in most cases IC will be manufactured in typical process case and will function in room temperature. Better speed performance of the chip in typical conditions allows reducing the supply voltage without impact on chip performance.

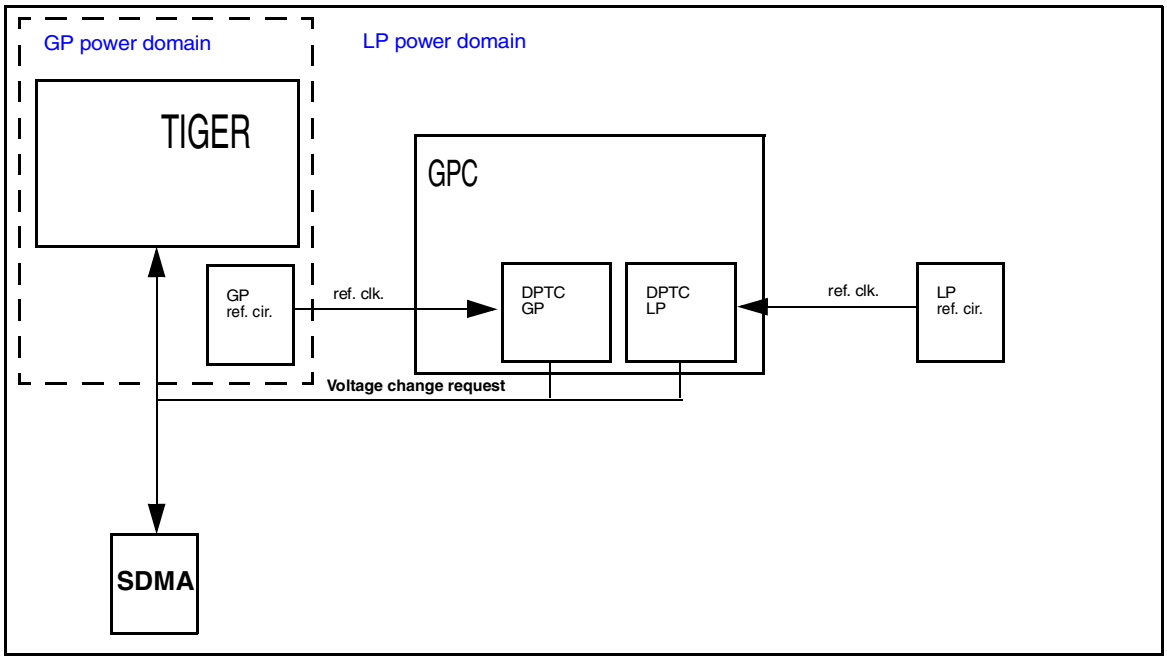
Considering different process flavors for CPU and peripheral power domains of the design, each of the domains will have dedicated DPTC engine, with relevant reference circuits connected.

Voltage update requests (to PMIC, via CSPI or using I2C) may be served in one of the following ways:

- Hardware triggering of CSPI, by GPC
- ARM interrupts serving CSPI
- ARM interrupts serving GPC int2 for I2C

The same engine, but with different reference circuits will be used to detect minimum opertaing voltage for logic part. The same can be used to define minimum operating or/and standby voltage for memory array .

Figure 34-9. DPTC System High Level Diagram



The DPTC spec describes the sequence of DPTC voltage change.

34.3.3 DVFS and DPTC Change Request Sequence Diagrams

Figure and Figure describe the sequence on DVFS or/and DPTC interrupt.

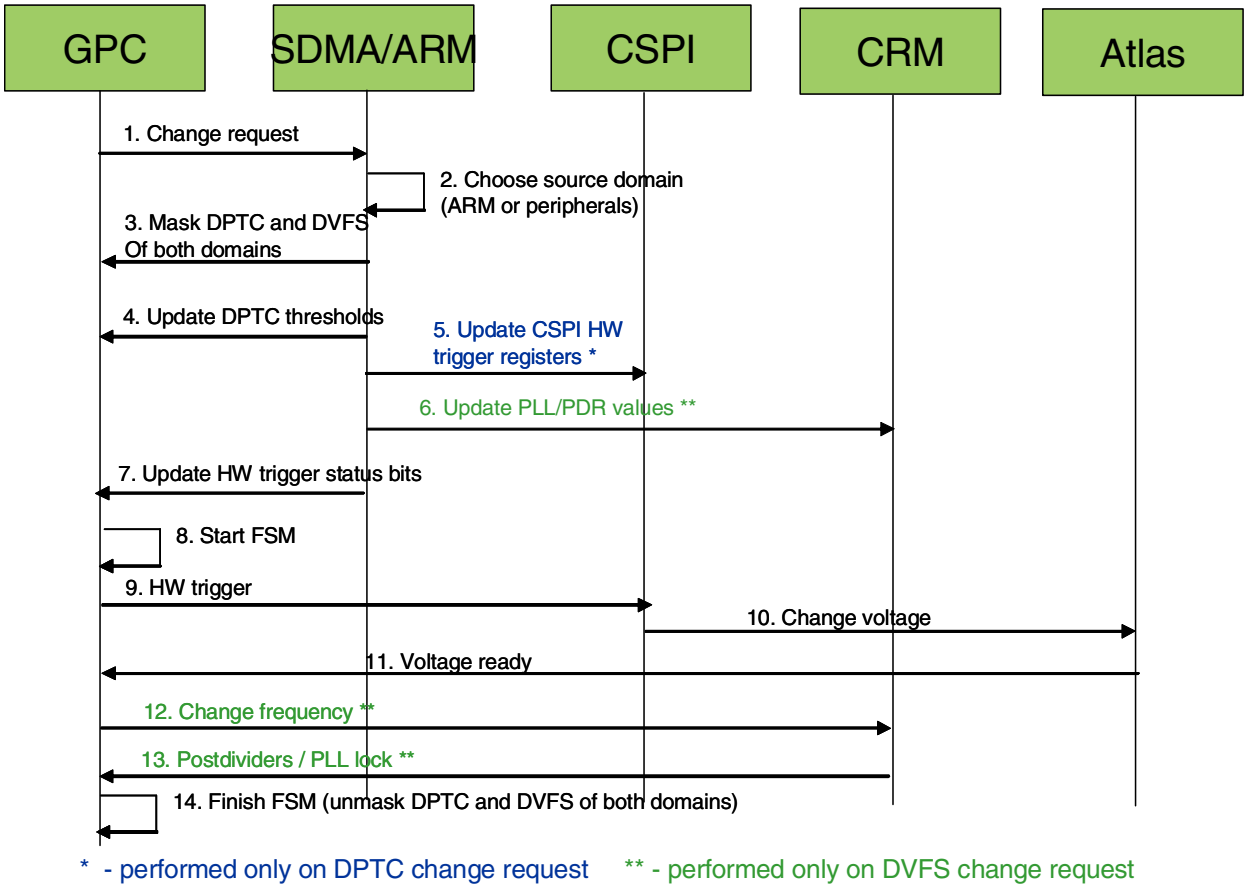


Figure 34-10. DVFS or/and DPTC Interrupt - Frequency Increase full_memories.csv

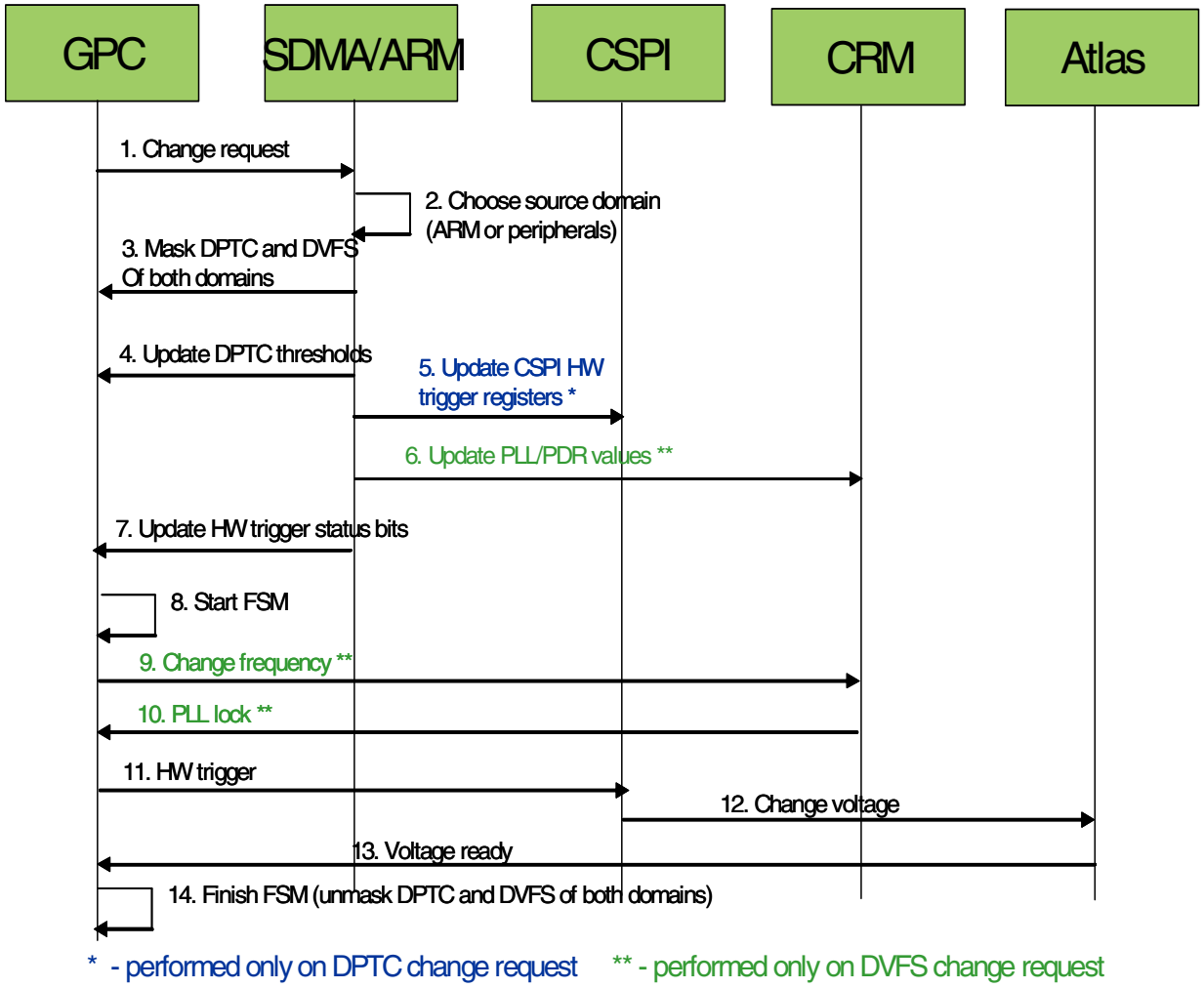


Figure 34-11. DVFS or/and DPTC Interrupt - frequency decrease

DPTC and DVFS modules of CPU and peripheral domains will have one common interrupt line to ARM and DMA request to SDMA. SDMA (or ARM) will choose only one domain to serve. If DPTC and/or DPTC interrupt of the other domainis pending, it will be triggered after the end of the first domain sequence.

Peripherals DVFS frequency change will be performed with post-dividers values update only.

CPU DVFS frequency change can be performed in 2 ways:

- PLL and post-dividers values update
- post-dividers values only update

When PLL update is required, Controller will switch CPU clock to another PLL, will wait until the previous PLL is locked on the required frequency, and then will switch the CPU clock back to the first PLL.

34.3.4 Frequency / Voltage change Controller description

Controller is responsible of correct frequency and voltage update flow.

34.3.4.1 GPC Controller Description

Table 34-12 and Table 34-10 describe the **GPC DVFS/DPTC state machine and the transition conditions.**

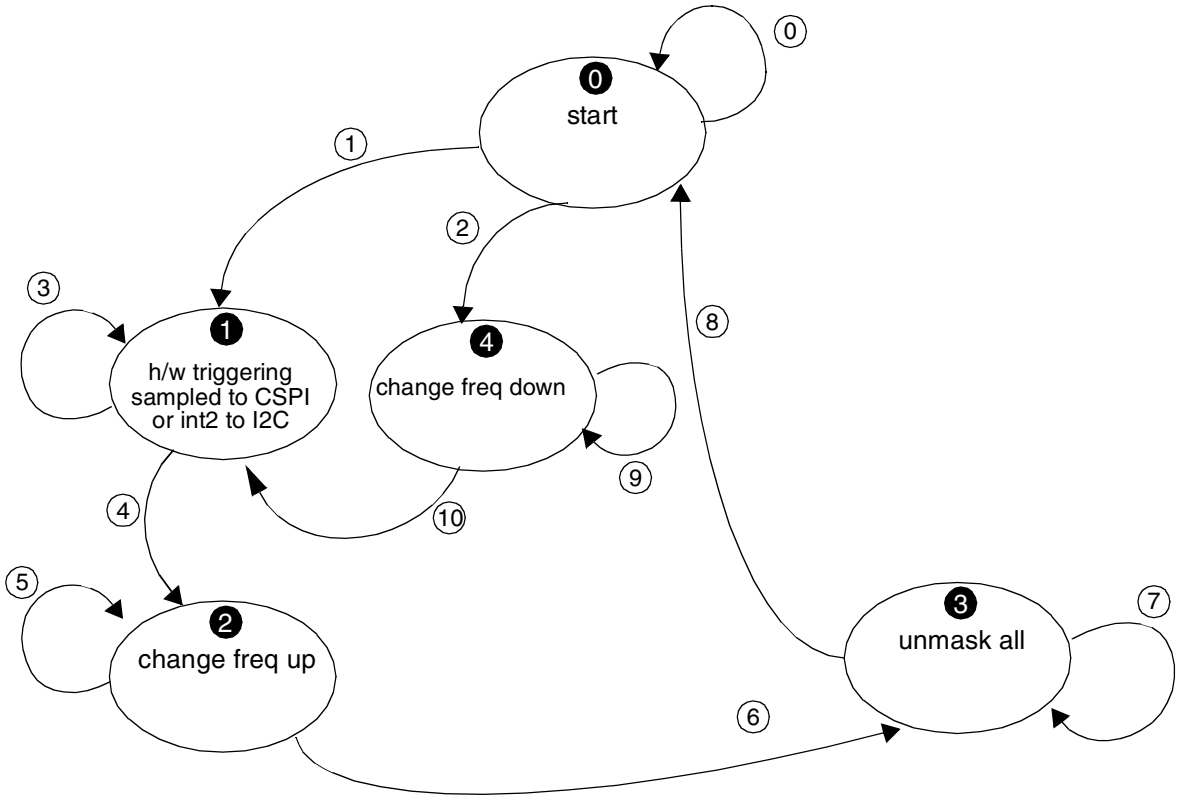


Figure 34-12. GPC DVFS/DPTC State machine

Table 34-10. GPC Transition Conditions

Transition	Transition condition	Output
0	not (1)	—

Table 34-10. GPC Transition Conditions

Transition	Transition condition	Output
1	STRT bit asserted, and ((freq/voltage increase needed) | (!freq/volt_inc & !freq_update))	—
2	STRT bit set, and freq/voltage decrease needed and freq_update	—
3	not (4)	1. sample h/w triggering selected reg index to CSPI or sending int2 to I2C for PMIC 2. Start counter if VCNTU==1 (pmic ack not needed)
4	(voltage_ready & pmic ack needed) | (counter finished & !pmic ack needed)	—
5	not (6)	1. if (dvfs update up), trigger freq. change 2. reset counter
6	(dvfs update up & clk change ack) | !(dvfs update up)	—
7	1 clk	unmask all, enable all toggle dvfs_cnt_res_arm or dvfs_cnt_res_per
8	not (8)	—
9	not (10)	1. if (dvfs update dw), trigger freq. change 2. IRQ masked: IRQM=1
10	(dvfs update dw& clk change ack) | !(dvfs update dw) | !freq_update	—

34.3.4.2 CCM Frequency Update Controller Description

Additional State Machine is required in CCM to take care about frequency update sequence, as desribed in Figure 34-13. Table 34-11 lists the CCM Frequency Update Controller transition conditions.

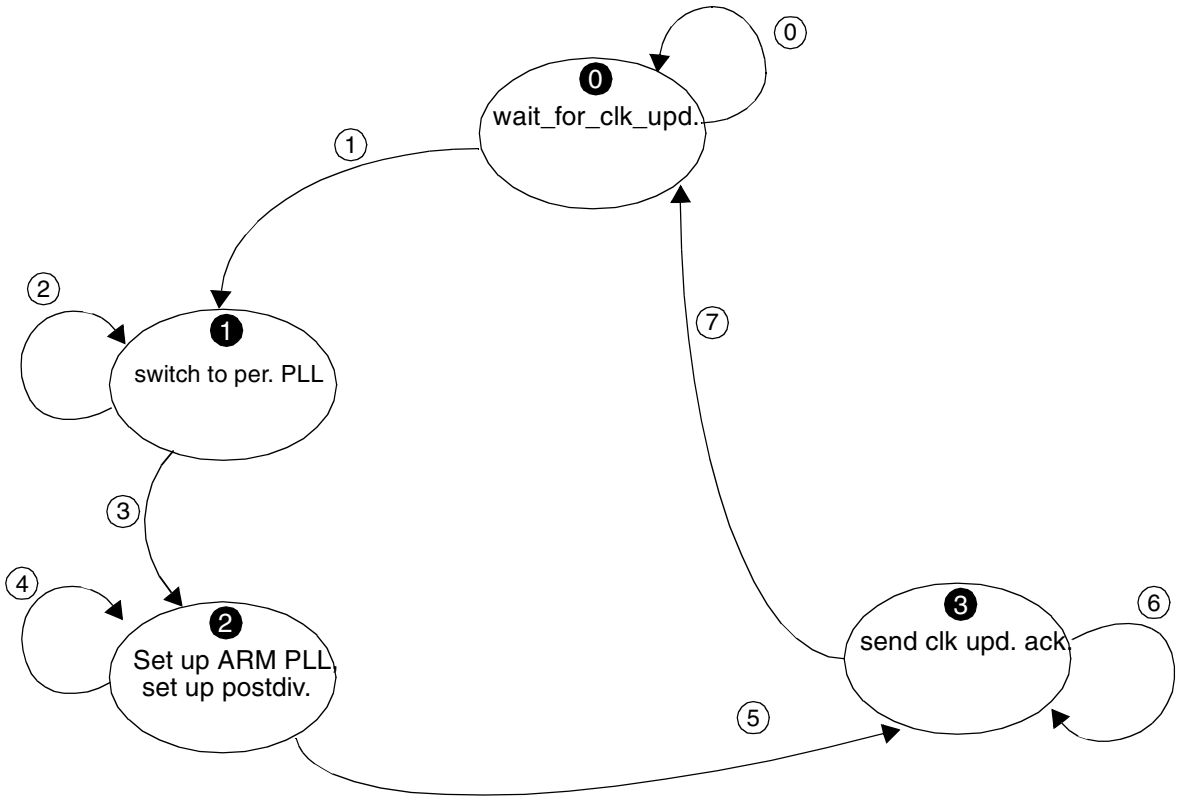


Figure 34-13. CCM Frequency Update Controller State Machine

Table 34-11. CCM Frequency Update Controller Transition Conditions

Transition	Transition condition	Output
0	not (1)	clk upd ack negated
1	clk update request	
2	not (3)	switch ARM clk to per. PLL
3	clk switched to per. PLL	
4	not (5)	apply new PLL settings, if needed apply new postdiv settings if needed
5	ARM PLL & postdiv upd ack.	
6	1 clk	clk update ack to GPC asserted.

Table 34-11. CCM Frequency Update Controller Transition Conditions

Transition	Transition condition	Output
7	not (6)	

34.3.5 State Retention Power Gating (SRPG)

State Retention Power Gating (SRPG) is an advanced power saving technique - the values of FFs are saved, while the supply for the combinational logic is gated. Such special power gating allows significant power saving during low-power (stand-by) states, when no logical operation is needed. Relatively the simple Power Gating techniques, SRPG allows to restore the state very quickly and continue the processing from the state, sampled before stand-by period.

SRPG Controller manages all the related signals for entering and exiting SRPG state of relevant modules. Not all the modules have SRPG design.

SRPG power down sequence:

All the modules except MEGAMIX get power down request, only after all the modules acknowledge received, and only if CTA8 was power downed, MEGAMIX receives power down request. The power to MEGAMIX is turned off only after the lock ready flags of all three PLLs are down.

SRPG power up sequence:

The modules receive power up request in following order:

MEGAMIX, IPU, VPU, GPU, EMI, (NEON and CTA8).

See Chapter 11, “Power Management,” for more information.

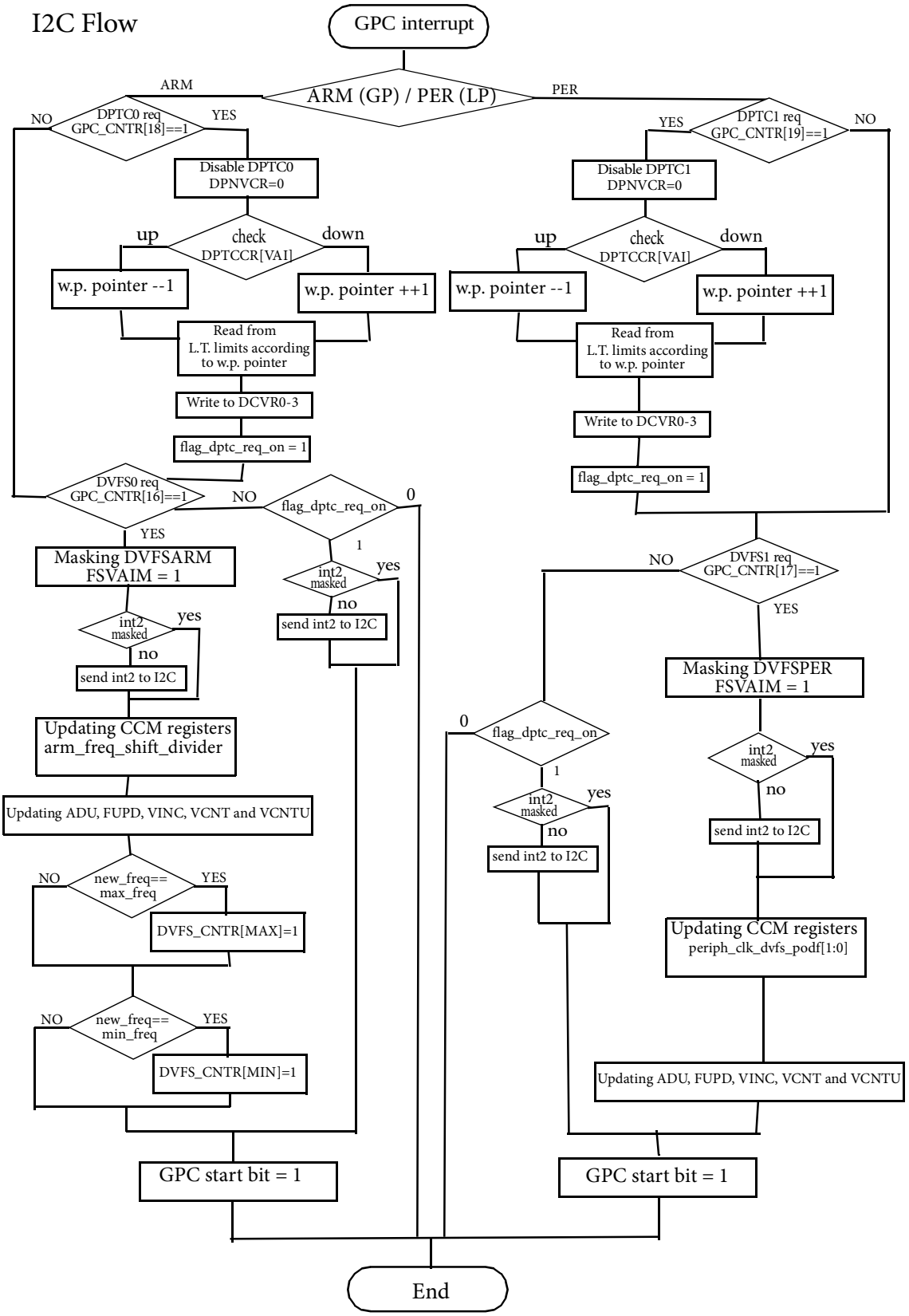
34.3.6 PMIC Interface Requirements for APM Support

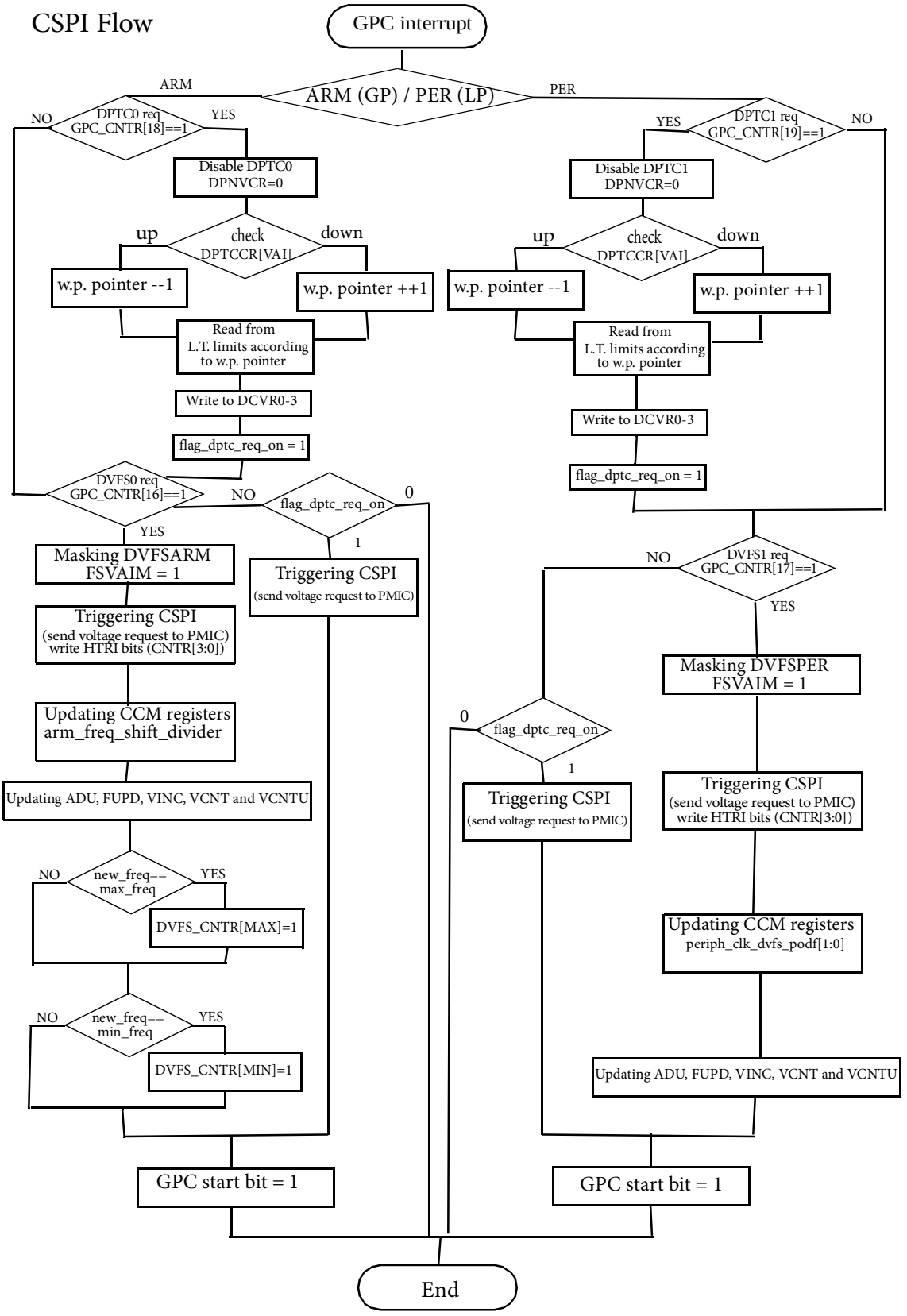
PMIC APM interface is done through SPI protocol by enhanced CSPI module with HW triggering. Below is the description of modules requirements for APM support.

- CSPI
  - Will have 16 HW trigger registers.
  - Each register will be 32 bits:
    - 1 bits read/write (always write)
    - 6 bits address
    - 1 bit not used
    - 24 bits data
  - Each register will have an associated signal that starts the transaction to Atlas
- SDMA/ARM
  - will update the values of the registers on DPTC interrupt for future DVFS frequencies (up to 14 on ARM domain, 2 on peripheral domain).

- GPC
  - will have a 4-bit select register to select one of 16 HW trigger signals, if CSPI is used.
  - will trigger the signals during DPTC/DVFS state machine according to select register value.
- PMIC
  - Only one 24-bit register should be updated for voltage change. One CSPI HW trigger will be able to make the change, or I2C will get int2.

Ability to use SDMA for h/w controllers inputs analysis and/or CSPI programming will provide high level of s/w flexibility and will be kept for backup.


I2C Flow
GPC interrupt
ARM (GP) / PER (LP)
ARM
PER
DPTC0 req GPC_CNTR[18]==1
NO
YES
Disable DPTC0 DPNVCR=0
check DPTCCR[VAI]
up
down
w.p. pointer --1
w.p. pointer ++1
Read from L.T. limits according to w.p. pointer
Write to DCVR0-3
flag_dptc_req_on = 1
DVFS0 req GPC_CNTR[16]==1
NO
YES
flag_dptc_req_on
0
1
Masking DVFSARM FSVAIM = 1
int2 masked
yes
no
send int2 to I2C
Updating CCM registers arm_freq_shift_divider
Updating ADU, FUPD, VINC, VCNT and VCNTU
new_freq== max_freq
NO
YES
DVFS_CNTR[MAX]=1
new_freq== min_freq
DVFS_CNTR[MIN]=1
GPC start bit = 1
DPTC1 req GPC_CNTR[19]==1
YES
NO
Disable DPTC1 DPNVCR=0
DVFS1 req GPC_CNTR[17]==1
Masking DVFSPER FSVAIM = 1
Updating CCM registers periph_clk_dvfs_podf[1:0]
End


CSPI Flow
GPC interrupt
ARM (GP) / PER (LP)
ARM
PER
DPTC0 req GPC_CNTR[18]==1
NO
YES
Disable DPTC0 DPNVCR=0
check DPTCCR[VAI]
up
down
w.p. pointer --1
w.p. pointer ++1
Read from L.T. limits according to w.p. pointer
Write to DCVR0-3
flag_dptc_req_on = 1
DVFS0 req GPC_CNTR[16]==1
flag_dptc_req_on
0
1
Masking DVFSARM FSVAIM = 1
Triggering CSPI (send voltage request to PMIC) write HTRI bits (CNTR[3:0])
Triggering CSPI (send voltage request to PMIC)
Updating CCM registers arm_freq_shift_divider
Updating ADU, FUPD, VINC, VCNT and VCNTU
new_freq== max_freq
DVFS_CNTR[MAX]=1
new_freq== min_freq
DVFS_CNTR[MIN]=1
GPC start bit = 1
DPTC1 req GPC_CNTR[19]==1
Disable DPTC1 DPNVCR=0
DVFS1 req GPC_CNTR[17]==1
Masking DVFSPER FSVAIM = 1
Updating CCM registers periph_clk_dvfs_podf[1:0]
End

Chapter 35
General Purpose Input/Output (GPIO)

The General Purpose Input/Output (GPIO) module provides 32 bits of bidirectional, general-purpose input and output signals. Figure 35-1 is a block diagram of the GPIO.

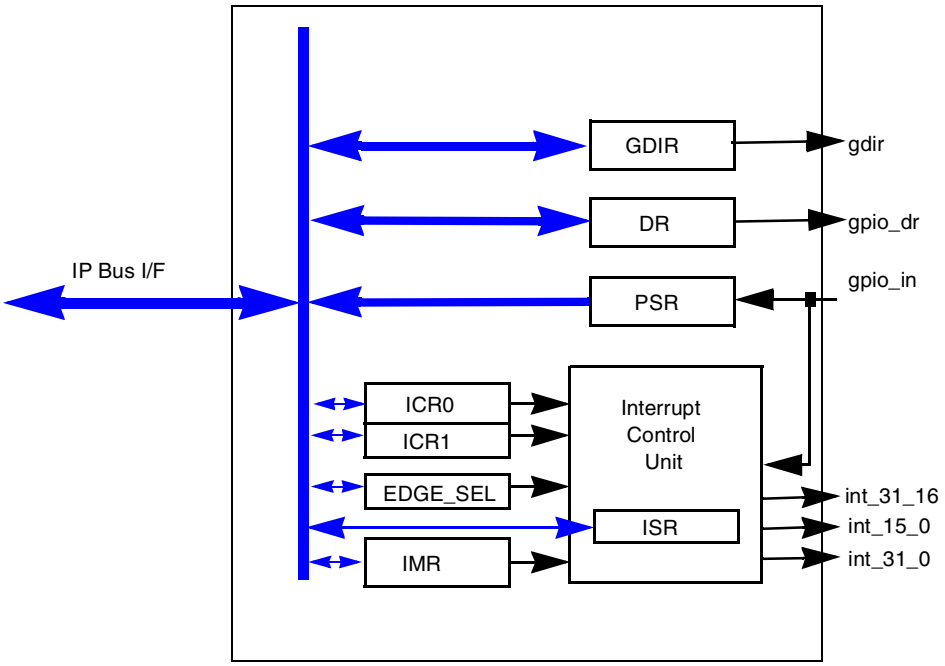


Figure 35-1. GPIO Block Diagram

35.1 Overview

The GPIO peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs. When GPIO is configured as an output, users can write to an internal register to control the state driven on the output pin. When GPIO is configured as an input, users can detect the state of the input by reading the state of an internal register.

The GPIO module is one of the modules controlling the IOMUX of the SoC (System on a Chip). Figure 35-2 shows the SoC muxing scheme.

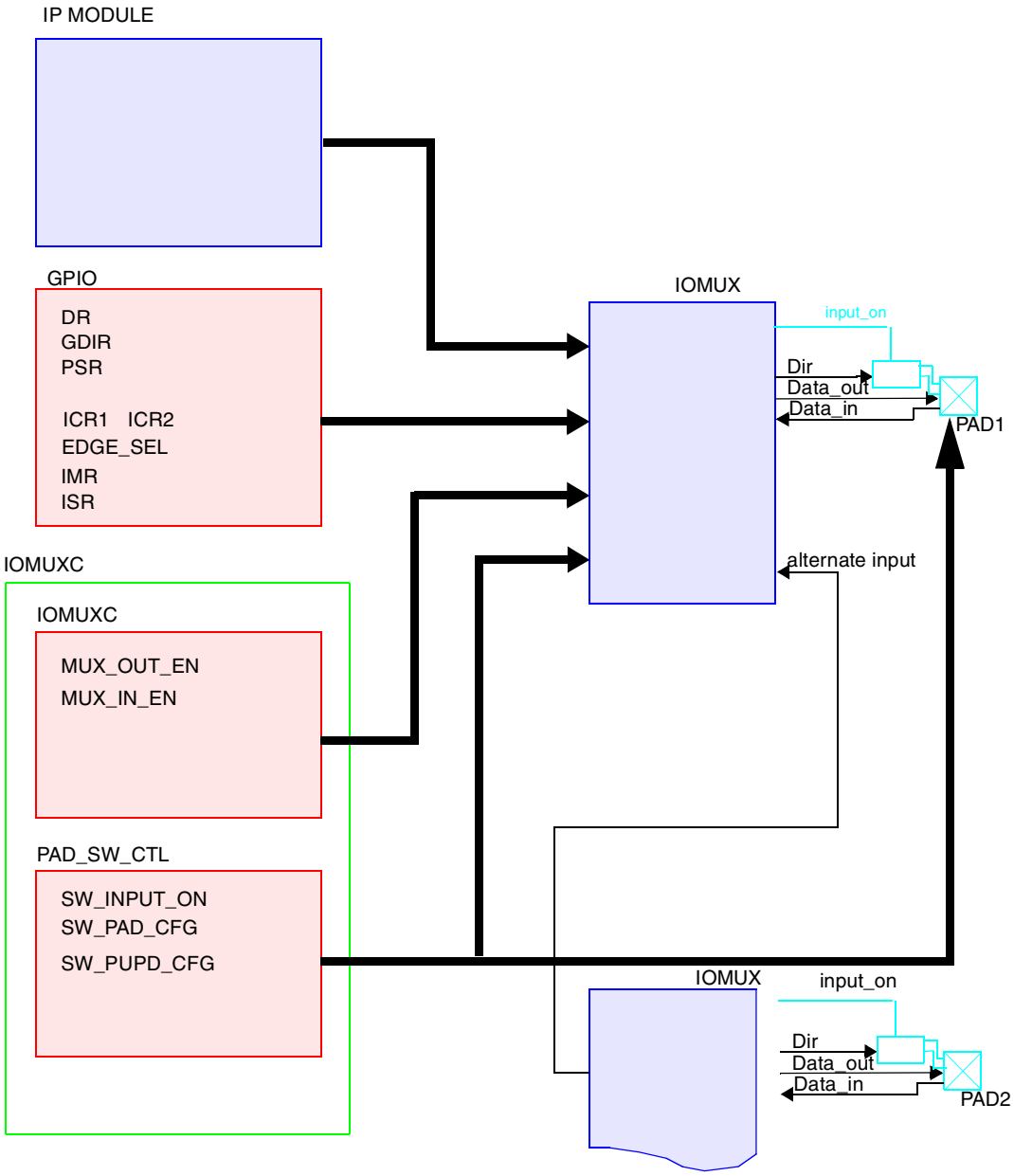


Figure 35-2. SoC IOMUX scheme

The functionality is provided through eight registers, an edge-detect circuit, and an interrupt generation logic.

The eight registers are as follows:

- DR—Data Register (32 bit)
- GDIR—Data Direction Register (32 bit)
- PSR—Pad Sample Register (32 bit)

- ICR (ICR1, ICR2)—Two Interrupt Control Registers (2 × 32 bit)
- EDGE_SEL—Edge Select register, selects edge detection.
- IMR—Interrupt Mask Register (32 bit)
- ISR—Interrupt Status Register (32 bit)

The data register is used to drive data from this register to I/O pins. Read access to the data register returns the value stored in the register or the value of the pad depending on the corresponding GDIR bit.

The data direction register controls the direction of the pin through the I/O multiplexer. Writing a one to this register configures the pin as an output, while a zero configures it as an input.

Each GPIO input has a dedicated edge-detect circuit that can be configured through software to detect rising edges, falling edges, a logic low-level, or a logic high-level on the input pin. Thirty two interrupts are supported. Two registers are used for this configuration, the interrupt configuration register (ICR1 and ICR2). Two bits are assigned to each GPIO pin, selecting one of the four possible detection methods. The EDGE_SEL register can override the ICR selection which forces edge detect (falling or rising edge).

The outputs of the edge detect circuits are optionally masked by setting the corresponding bit in the interrupt mask register (IMR). These qualified outputs are ORed together to generate three interrupt lines:

- interrupt_or_31_16 : 1One-bit interrupt OR of 16 high interrupts.
- interrupt_or_15_0 : 1One-bit interrupt OR of 16 low interrupts.
- ipi_gpio_int : One-bit interrupt OR of 32 interrupts.
- ipi_gpio_int32 : 32 bit, All Interrupts Out.

35.1.1 Features

The GPIO includes the following features:

- General purpose input/output logic supports the following:
  - Ability to drive a specific data to the pad using DR register
  - Ability to control the direction of the pad using the GDIR register
  - The core has the ability to sample the status of the corresponding pads by reading the PSR register.
- GPIO interrupts support the following:
  - Ability to support up to 32 interrupts
  - Ability to identify interrupt edges
  - Generation of three active high interrupts to the SoC interrupt controller

35.2 External Signal Description

In addition to the IP bus interface, the GPIO has the signals listed in Table 35-1.

Table 35-1. Signal Properties

Name	Port	Function	Reset	Pull Up
ipp_do_gpio_gdir	—	GPIO direction. This is the GPIO direction signal that controls the direction of the pad if the IOMUX is in GPIO mode. 0 GPIO is configured as input. 1 GPIO is configured as output.	0	—
ipp_do_gpio_dr	—	GPIO Data Register. This is the GPIO data signal. If the IOMUX is in GPIO mode and the corresponding direction bit is set to output, this signal will be driven to the pad.	0	—
ipp_ind_g_in	—	GPIO Input. This is the input line coming from the IOMUX, so the core can read the status of the pad.	0	—
ipi_gpio_int31_16	—	GPIO outputs functioning as interrupt—MSB ORed interrupts. One-bit interrupt OR of 16 high interrupts.	0	—
ipi_gpio_int15_0	—	GPIO outputs functioning as interrupt—LSB ORed interrupts. One-bit interrupt OR of 16 low interrupts.	0	—
ipi_gpio_int	—	GPIO outputs functioning as interrupt—ORed interrupts. One-bit interrupt OR of 32 interrupts.	0	—
ipi_gpio_int32	—	GPIO outputs functioning as interrupt—32-bit, All Interrupt Out.	0	—
ipg_clk_s	—	The only input clock—ipg_clk, gated by the module module_en signal. The clock is working only when GPIO is accessed.	—	—
ips_rdata	—	Read data bus.	—	—
ips_xfr_wait	—	Wait cycle. This signal indicates GPIO does not wish to complete transfer in the current cycle.	—	—
$\overline{\text{ipg\_hard\_async\_reset}}$	—	Hardware reset signal.	—	—
ips_module_en	—	Module enable input.	—	—
ips_addr	—	IP address bus.	—	—
ips_wdata	—	Write data bus. Data to write into target	—	—
ips_rwb	—	This signal is the read/write control signal. This signal Indicates a read transfer when asserted and a write transfer when negated.	—	—
ips_byte_31_24	—	These signals are byte enables 31 to 24 only—32-bit access available.	—	—
ips_byte_23_16	—	These signals are byte enables 23 to 16 only—32-bit access available.	—	—
ips_byte_15_8	—	These signals are byte enables 15 to 8 only—32-bit access available.	—	—
ips_byte_7_0	—	These signals are byte enables 7 to 0 only—32-bit access available.	—	—
ipt_test_mode	—	This signal is the scan mode signal.	—	—
ipt_test_async_se	—	This signal enables the bypass reset.	—	—
ipt_test_clk_se	—	This signal enables the clock (clk) in scan mode.	—	—

35.3 Memory Map and Register Definition

There are eight GPIO registers. All registers are accessible from the IP interface. Only 32-bit access is supported.

35.3.1 Memory Map

Table 35-2 shows the GPIO memory map.

Table 35-2. GPIO Memory Map [1]

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x000 (DR)	GPIO Data	R/W	0X0000_0000	35.3.3.1/35-7
0xBASE+0x004 (GDIR)	GPIO Direction	R/W	0X0000_0000	35.3.3.2/35-8
0xBASE+0x008 (PSR)	GPIO Pad Status	Read-only	0X0000_0000	35.3.3.3/35-9
0xBASE+0x00C (ICR1)	GPIO Interrupt Configuration Register1	R/W	0X0000_0000	35.3.3.4/35-10
0xBASE+0x010 (ICR2)	GPIO Interrupt Configuration Register2	R/W	0X0000_0000	35.3.3.5/35-10
0xBASE+0x014 (IMR)	GPIO Interrupt Mask Register	R/W	0X0000_0000	35.3.3.6/35-11
0xBASE+0x018 (ISR)	GPIO Interrupt Status Register	R/W	0X0000_0000	35.3.3.7/35-11
0xBASE+0x01C	GPIO Edge Detect Register	R/W	0X0000_0000	

[1] See Chapter 2, "Memory Map," for the value of base address of GPIO.

35.3.2 Register Summary

The following definitions serve as a key for the WMSG register summary and individual register diagrams.

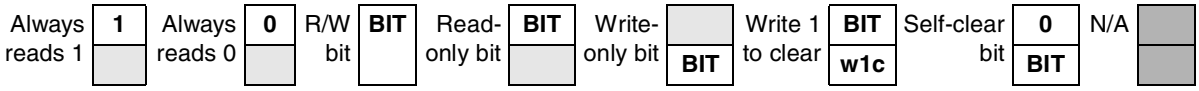


Figure 35-3. Key to Register Fields

Table 35-3 provides a key for register figures and the register summary table.

Table 35-3. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).

Table 35-3. Register Conventions (continued)

Convention	Description
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero (previously labeled slfclr).
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 35-4 shows the GPIO register summary.

Table 35-4. GPIO Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (DR)	R	DR[31:16]															
	W																
	R	DR[15:0]															
	W																
0xBASE+0x004 (GDIR)	R	GDIR[31:16]															
	W																
	R	GDIR[15:0]															
	W																
0xBASE+0x008 (PSR)	R	PSR[31:16]															
	W																
	R	PSR[15:0]															
	W																
0xBASE+0x00C (ICR1)	R	ICR1[31:16]															
	W																
	R	ICR1[15:0]															
	W																

Table 35-4. GPIO Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x010 (ICR2)	R	ICR2[31:16]															
	W																
	R	ICR2[15:0]															
	W																
0xBASE+0x014 (IMR)	R	IMR[31:16]															
	W																
	R	IMR[15:0]															
	W																
0xBASE+0x018 (ISR)	R	ISR[31:16]															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	ISR[15:0]															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
EDGE_SEL 0xBASE+0x01C	R	EDGE_SEL[31:16]															
	W																
	R	EDGE_SEL[15:0]															
	W																

35.3.3 Register Descriptions

This section contains the detailed register descriptions for the GPIO registers.

35.3.3.1 GPIO Data Register (DR)

The GPIO DR register is a 32-bit register. Each bit stores a data to be driven to the pad at all times. If the IOMUX is in GPIO mode and the direction is output, this data will be driven there. If the direction is input, a read action to DR bit reflects the value on the corresponded pad. Two wait states are required in read access for synchronization.

The data returned when reading the DR register is a function of the IOMUX input mode settings and the corresponding GDIR bit, as follows.

- If GDIR == 1 && IOMUX input mode == GPIO, reading DR returns the content of the DR register
- If GDIR == 0 && IOMUX input mode == GPIO, reading DR returns the pad's value
- If GDIR == 1 && IOMUX input mode != GPIO, reading DR returns the content of the DR register
- If GDIR == 0 && IOMUX input mode != GPIO, reading DR returns zero.

Figure 35-4 shows the DR register and Table 35-5 shows the register's field descriptions.

0xBASE+0x000 (DR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DR[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 35-4. GPIO Data Register (DR)

Table 35-5. DR Field Descriptions

Field	Description
31–0 DR	Data bits. This register defines the value of the GPIO output when the pin is configured as an output (GDIR[n]=1). Writes to this register are stored in a register. Reading DR returns the value stored in the register if the pin is configured as an output (GDIR=1), or the state of the I/O pin if configured as an input (GDIR[n]=0). Settings: The I/O multiplexer associated with each bit must be configured for GPIO for the function to affect the state of the pin. Reading the data register with the input path disabled will always return a zero value.

35.3.3.2 GPIO Direction Register (GDIR)

The GPIO GDIR register is a 32-bit register that functions as direction control when the IOMUX direction is controlled by this bit. Each bit specifies the direction of a specific pad. The mapping of each DIR bit to a corresponding pad is determined on the SoC's pin assignment and IOMUX table. Figure 35-5 shows the GDIR register and Table 35-6 shows the register's field descriptions.

0xBASE+0x004 (GDIR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GDIR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GDIR[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 35-5. GPIO Direction Register (GDIR)

Table 35-6. GDIR Field Descriptions

Field	Description
31–0 GDIR	GPIO Direction bits. Bit i of this register defines the direction of the GPIO[i] signal. 0 GPIO is configured as input. 1 GPIO is configured as output. **Note:** GDIR affects only the direction of the I/O pin when the corresponding bit in the I/O MUX is configured for GPIO.

35.3.3.3 GPIO Pad Status Register (PSR)

The GPIO PSR register is a 32-bit read-only register. Each bit stores the value of the corresponding pad. This register is clocked with the ipg_clk_s clock, meaning that the value on the pad is sampled only when accessing this location. Two wait states are required any time this register is accessed for synchronization. Figure 35-6 shows the PSR register and Table 35-7 shows the register's field descriptions.

NOTE

PSR[i]—pad sample

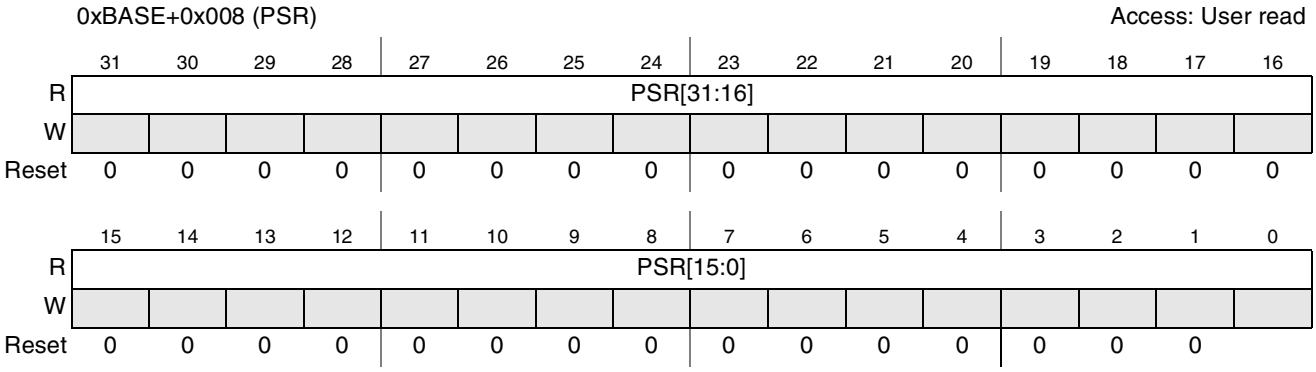


Figure 35-6. GPIO Pad Status Register (PSR)

Table 35-7. PSR Field Descriptions

Field	Description
31–0 PSR	GPIO Pad Status bits (status bits). Reading PSR returns the state of the corresponding pad. Settings: The I/O multiplexer associated with each bit must be configured for GPIO for the function to affect the state of the pin.

35.3.3.4 GPIO Interrupt Configuration Register1 (ICR1)

The GPIO ICR1 register is a 32-bit register. Each set of 2 bits specifies the interrupt configuration for each corresponding interrupt line. There is total support for 16 interrupts. Figure 35-7 shows the ICR1 register, and Table 35-8 shows the register's field descriptions.

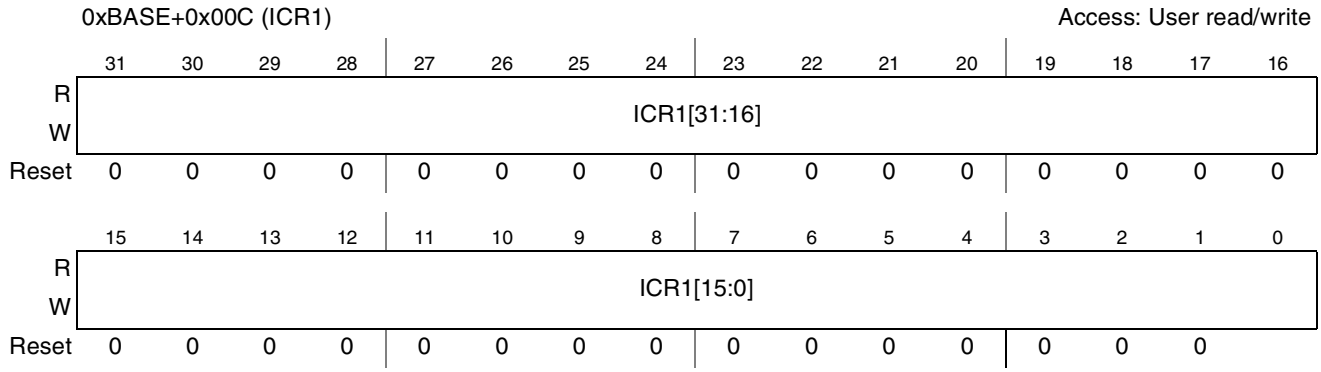


Figure 35-7. GPIO Interrupt Configuration Register1 (ICR1)

Table 35-8. ICR1 Field Descriptions

Field	Description
31–0 ICR1	Interrupt Configuration 1 bits. This register controls the active condition of the interrupt function for lines 15 to 0. Settings: interrupts (i) 0 to 15, when bits {ICR1[2*i+1],ICR1[2*i]} are: 00 The interrupt i is low-level sensitive. 01 The interrupt i is high-level sensitive. 10 The interrupt i is rise-edge sensitive. 11 The interrupt i is fall-edge sensitive.

35.3.3.5 GPIO Interrupt Configuration Register2 (ICR2)

The GPIO ICR2 register is a 32-bit register. Each set of 2 bits specifies the interrupt configuration for each corresponding interrupt line. There is total support for 16 interrupts. Figure 35-8 shows the ICR2 register and Table 35-9 shows the register's field descriptions.

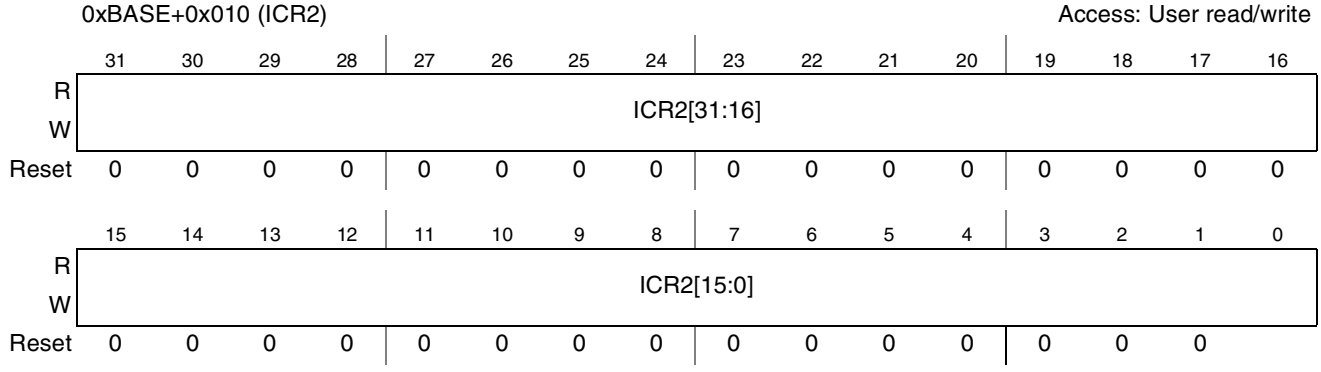


Figure 35-8. GPIO Interrupt Configuration Register2 (ICR2)

Table 35-9. ICR2 Field Descriptions

Field	Description
31–0 ICR2	Interrupt Configuration 2 bits. This register controls the active condition of the interrupt function for lines 31 to 15. Settings: interrupts (i) 0 to 15, when bits {ICR2[2*i+1],ICR2[2*i]} are: 00 The interrupt i+16 is low-level sensitive. 01 The interrupt i+16 is high-level sensitive. 10 The interrupt i+16 is rise-edge sensitive. 11 The interrupt i+16 is fall-edge sensitive.

35.3.3.6 GPIO Interrupt Mask Register (IMR)

The GPIO IMR is a 32 bit register. Each bit is the interrupt masking bit for each interrupt line. Figure 35-9 shows the IMR register and Table 35-10 shows the register's field descriptions.

0xBASE+0x014 (IMR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IMR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IMR[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 35-9. GPIO Interrupt Mask Register (IMR)

Table 35-10. IMR Field Descriptions

Field	Description
31–0 IMR	Interrupt Mask bits. This register is used to enable/disable the interrupt function on each of the 32 GPIO pins. Settings: For i from 0 to 31, when IMR[i] is: 0 The interrupt i is disabled. 1 The interrupt i is enabled.

35.3.3.7 GPIO Interrupt Status Register (ISR)

The GPIO ISR is a 32- bit register that functions as interrupt. Each bit indicates whether an interrupt has occurred. When an interrupt event occurs, the bit in this register is set. The condition for setting of the bit is determined by the Interrupt Configuration Register (ICR) and the input that satisfies the configuration.

Two wait states are required in read access for synchronization. One wait state is required for reset. Figure 35-10 shows the ISR register and Table 35-11 shows the register's field descriptions.

0xBASE+0x018 (ISR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ISR[31:16]															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR[15:0]															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 35-10. GPIO Interrupt Status Register (ISR)

Table 35-11. ISR Field Descriptions

Field	Description
31–0 IMR	Interrupt Status bits — Bit i of this register is asserted (active high) when the active condition is detected on the GPIO input and is waiting for service. The value of this register is independent of the value in the IMR register. When the active condition has been detected, the corresponding bit remains set until cleared by software. Status flags are cleared by writing a 1 to the corresponding bit position.

35.3.3.8 GPIO Edge Select Register (EDGE_SEL)

The GPIO EDGE_SEL register is a 32-bit register. There is total support for 32 interrupts. Figure 35-8 shows the EDGE_SEL register and Table 35-9 shows the register's field descriptions. This register overrides the ICR registers configuration to select edge detection rising edge or falling edge. This register was added for backward compatibility reasons, its reset value is set to disregard its functionality.

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDGE_SEL[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDGE_SEL[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 35-11. GPIO Interrupt Configuration Register2 (ICR2)

Table 35-12. ICR2 Field Descriptions

Field	Description
31–0 EDGE_SEL	When bit[n] is set, the GPIO disregards ICR functionality and detects any edge on the corresponding interrupt input bit.

35.4 Functional Description

This section describes the GPIO function and programming.

35.4.1 GPIO Function

A GPIO pin can operate as a general-purpose input/output when the IOMUX functions in GPIO mode. Each GPIO pin can be independently configured as either an input or an output using the GPIO Direction Register (GDIR). When configured as an output (GDIR bit = 1), the value in the data bit in the GPIO Data Register (DR) is driven on the corresponding GPn pin. When configured as an input (GDIR bit = 0), the state of the input can be read from the corresponding PSR bit.

35.4.2 GPIO Programming

This section explains the programming sequence for reading and writing values from pad.

35.4.2.1 Read Value from Pad

The programming sequence should be as follows:

1. Configure IOMUX to select GPIO mode (Via IOMUXC).
2. Configure GPIO Direction Register to input.
3. Read value from Data register/Pad Status register.

The pseudocode description for reading [pad3:pad0] values is as follows:

```
write sw_mux_ctl_<pad0>_<pad1>_<pad2>_<pad3> , 32'h00000000          // SET PADS TO GPIO MODE.
write GDIR[31:4,pad3_bit,pad2_bit, pad1_bit, pad0_bit,] 32'hxxxxxxx0 // SET GDIR TO INPUT.
read DR                                                              // READ PAD VALUE FROM DR.
read PSR                                                             // READ PAD VALUE FROM PSR.
```

NOTE

While GPIO direction is set to input (GDIR = 0), a read access to DR does not return DR data. Instead, it returns the PSR data, which is the corresponding pad value.

35.4.2.2 Write Value to Pad

The programming sequence should be as follows:

1. Configure IOMUXto select GPIO mode (Via IOMUXC).

2. Configure GPIO Direction Register to output.
3. Write value to Data Register (DR).

The pseudocode description to drive 4’b0101 on [pad3:pad0] is as follows:

```
write sw_mux_ctl_<pad0>_<pad1>_<pad2>_<pad3> , 32’h00000000          // SET PADS TO GPIO MODE.
write GDIR[31:4,pad3_bit,pad2_bit, pad1_bit, pad0_bit,] 32’hxxxxxxxF // SET GDIR TO OUTPUT.
write DR, 32’hxxxxxxx5                                               // WRITE PAD VALUE TO DR.
read_cmp PSR, 32’hxxxxxxx5                                     // READ PAD VALUE FROM PSR ONLY.
```

NOTE

While GPIO direction is set to output, real pad value can only be verified through PSR.

35.4.3 Interrupt Control Unit

In addition to the general-purpose input/output function, the edge-detect logic in the GPIO peripheral reflects whether a transition has occurred on a given GPIO signal that is configured as an input (GDIR bit = 0). The GPIO signal transition is reflected in the GPIO ICR registers. The GPIO ICR registers enables to configure each interrupt input to its sensitivity case (low-to-high transition; high-to-low transition; low; high).

The interrupt control unit is built of 32 interrupt control sub units. Each sub unit handles a single interrupt line.

.

Chapter 36
General Purpose Timer (GPT)

36.1 Introduction

A block diagram of the General Purpose Timer (GPT) outlining its basic functionality is shown in Figure 36-1.

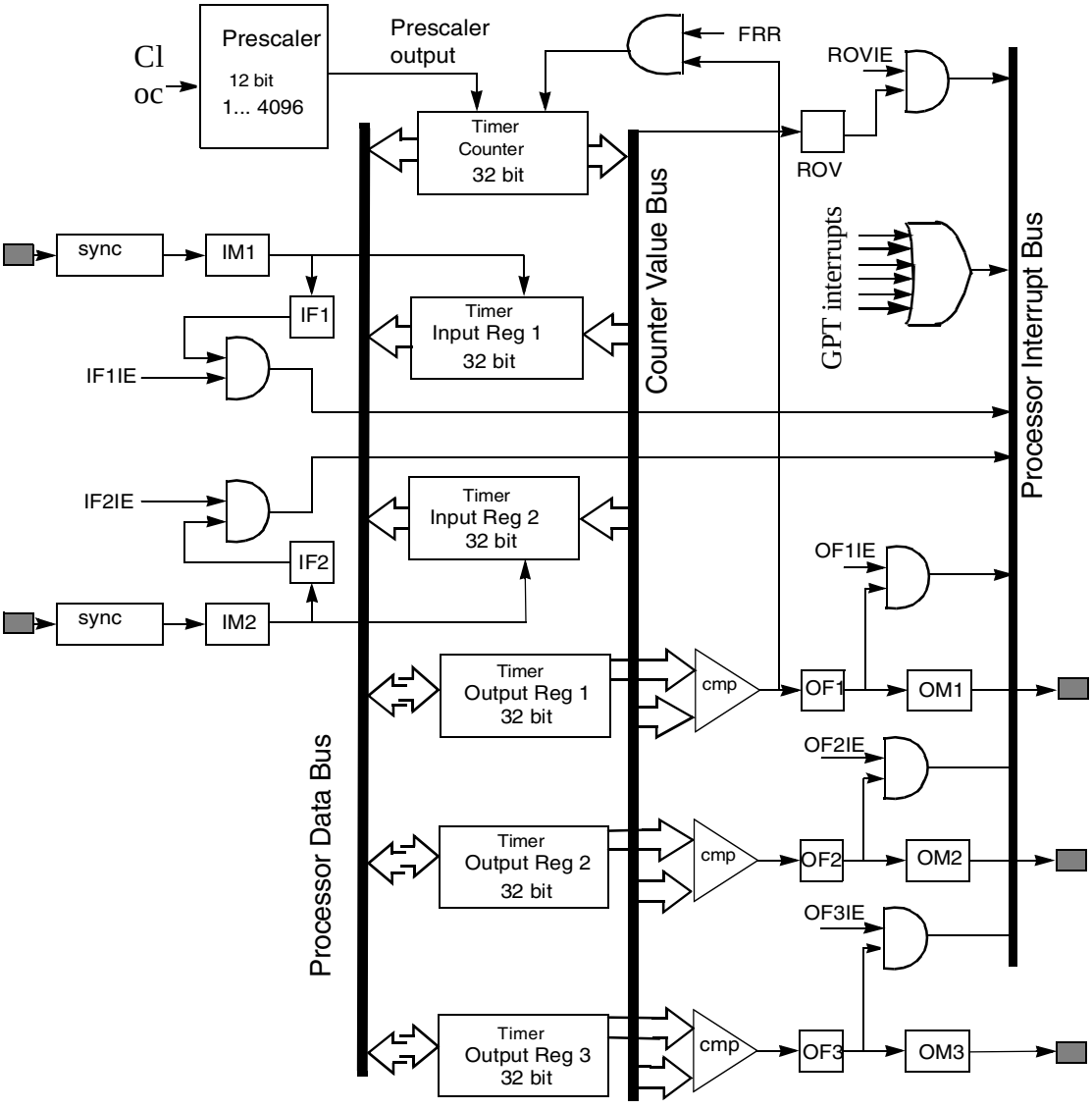


Figure 36-1. General Purpose Timer (GPT) Block Diagram

Figure 36-2 shows the GPT functional clocking scheme

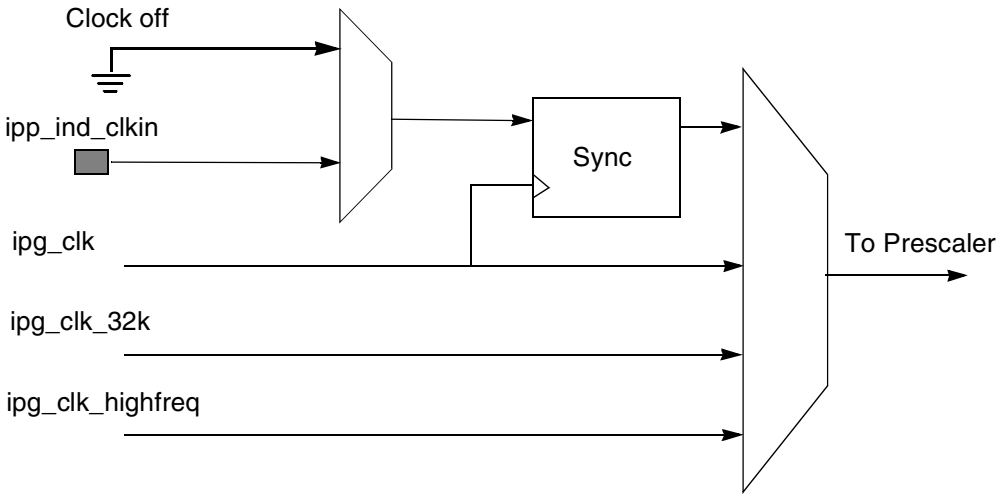


Figure 36-2. GPT Counter Clocks Diagram

36.1.1 Overview

The General purpose timer (GPT) has a 32 bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge. The GPT can also generate an event on ipp_do_cmpout pins and an interrupt when the timer reaches a programmed value. It has a 12 bit prescaler providing a programmable clock frequency derived from multiple clock sources.

36.1.2 Features

The features are as follows:

- One 32 bit up-counter with clock source selection, including external clock.
- Two input capture channels with programmable trigger edge.
- Three output compare channels with programmable output mode. Forced compare feature also available.
- Can be programmed to be active in low power and debug modes
- Interrupt generation at capture, compare, rollover events.
- Restart or free-run modes for counter operation.

36.1.3 Modes of Operation

The GPT can be programmed to run in Free-run or Restart modes.

In restart mode, (selectable through control register), compare channel 1 behaves differently than the other two compare channels. For compare channel 1, every time the counter reaches the compared value, it resets and starts again from 0x0000_0000. Any write access to the compare register of channel 1 results in the GPT counter being reset. This is to avoid possible missing of compare event when compare value is

changed from a higher to lower value while counting is going on. For the other two compare channels, the compare event occurs, but the counter is not reset.

In free-run mode, the compare event occurs for all three channels, and the counter value is not reset. The counter continues to count until 0xFFFF_FFFF and rolls over to 0x0000_0000.

36.2 Signal Description

The GPT follows IP Bus protocol for interfacing with the Processor core. It does not have any interface signals with any other module inside the chip except for the clock and reset inputs from the Clock and Reset Controller module and the interrupt signals to the processor interrupt handler.

There are functional and clock inputs and functional output signals going outside the chip boundary. Table 36-1 outlines these signals.

Table 36-1. External Signal Description

Name	Direction	Function	Reset State	Pull up
ipp_ind_clkin	Input	Input pin for external clock on which counter can be run	—	Passive Hysteresis
ipp_ind_capin1	Input	Input pin for capture event for capture channel 1	—	Passive
ipp_ind_capin2	Input	Input pin for capture event for capture channel 2	—	Passive
ipp_do_cmpout1	Output	Output pin for indication of compare event occurrence in compare channel 1	0	Passive
ipp_do_cmpout2	Output	Output pin for indication of compare event occurrence in compare channel 2	0	Passive
ipp_do_cmpout3	Output	Output pin for indication of compare event occurrence in compare channel 3	0	Passive

36.2.1 External Signals

This section describes the six signals (three input and three output) in the GPT module that are to be connected to the chip pads.

36.2.2 ipp_ind_clkin — External Clock Input

The GPT counter can be run on external clock from outside the chip and this is the input pin on which this clock is available. This clock is treated as asynchronous to ipg_clk. Its frequency should be less than 1/4 of frequency of ipg_clk for proper functioning of GPT. Hysteresis characteristics on this pad will be required as this is a clock input.

36.2.3 ipp_ind_capin1, ipp_ind_capin2 — Input Capture Trigger Signals

The GPT counter value can be stored into a register with an event from outside the chip. A positive or/and negative edge on these signals can trigger this capture event. These signals are treated as asynchronous to

ipg_clk. Only those transitions which occur at least a single clock cycle (clock selected to run the counter) after the previous recorded transition will be guaranteed to trigger a capture event.

36.2.4 ipp_do_cmpout1, ipp_do_cmpout2, ipp_do_cmpout3—Output Compare Signals

These signals show the occurrence of output compare event through a specified transition.

36.3 Register Definition and Memory Map

The GPT has 10 user-accessible 32-bit registers. These registers are used to configure, operate, and monitor the state of the GPT.

IP Bus Write access to GPT Control Register (GPTCR) and GPT Output Compare Register1 (GPTOCR1) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while other valid IP bus accesses are with 0 wait state.

Irrespective of resp_sel signal value, Write access to GPT Status Registers (Read only registers GPTICR1, GPTICR2, GPTCNT) will generate the bus exception (ips_xfr_err signal will be asserted). If resp_sel is driven low then Read/Write access to the unimplemented address space of GPT (ips_addr greater than or equal to $BASE + $028) will generate the bus exception (ips_xfr_err signal will be asserted). If resp_sel is driven high then Read/Write access to the unimplemented address space of GPT will not create any error response.

Table 36-2 shows the register memory map.

Table 36-2. Block Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x0000 (GPTCR)	GPT Control Register	32-bit R/W	0x0000_0000	36.3.2.1/36-6
0xBASE+0x0004 (GPTPR)	GPT Prescaler Register	32-bit R/W	0x0000_0000	36.3.2.2/36-10
0xBASE+0x0008 (GPTSR)	GPT Status Register	32-bit R/W	0x0000_0000	36.3.2.3/36-10
0xBASE+0x000c (GPTIR)	GPT Interrupt Register	32-bit R/W	0x0000_0000	36.3.2.4/36-11
0xBASE+0x0010 (GPTOCR1)	GPT Output Compare Register 1	32-bit R/W	0xFFFF_FFFF	36.3.2.5/36-12
0xBASE+0x0014 (GPTOCR2)	GPT Output Compare Register 2	32-bit R/W	0xFFFF_FFFF	36.3.2.6/36-13
0xBASE+0x0018 (GPTOCR3)	GPT Output Compare Register 3	32-bit R/W	0xFFFF_FFFF	36.3.2.7/36-13
0xBASE+0x001c (GPTICR1)	GPT Input capture Register 1	32-bit Read	0x0000_0000	36.3.2.8/36-14
0xBASE+0x0020 (GPTICR2)	GPT Input capture Register 2	32-bit Read	0x0000_0000	36.3.2.9/36-14
0xBASE+0x0024 (GPTCNT)	GPT Counter Register	32-bit Read	0x0000_0000	36.3.2.10/36-15

36.3.1 Register Summary

Table 36-3 summarizes all the GPT registers.

KEY:

Always Reads One	Always Reads Zero	Read/ Write Bit	Read-Only Bit	Write-Only Bit	Write 1 to Clear	Self-Clear Bit	N/A
1	0	bit	bit	bit	bit	0	
					w1c	bit	

Table 36-3. GPT Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0004 (GPTPR)	R																
	W																
	R	0	0	0	0	PRESCALER[11:0]											
	W																
0xBASE+0x0008 (GPTSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	ROV	IF2	IF1	OF3	OF2	OF1
	W											w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x000c (GPTIR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	ROVIE	IF2IE	IF1IE	OF3IE	OF2IE	OF1IE
	W																
0xBASE+0x0010 (GPTOCR1)	R	COMP[31:16]															
	W																
	R	COMP[15:0]															
	W																
0xBASE+0x0014 (GPTOCR2)	R	COMP[31:16]															
	W																
	R	COMP[15:0]															
	W																
0xBASE+0x0018 (GPTOCR3)	R	COMP[31:16]															
	W																
	R	COMP[15:0]															
	W																
0xBASE+0x001c (GPTICR1)	R	CAPT[31:16]															
	W																
	R	CAPT[15:0]															
	W																
0xBASE+0x0020 (GPTICR2)	R	CAPT[31:16]															
	W																
	R	CAPT[15:0]															
	W																

Table 36-3. GPT Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0024 (GPTCNT)	R	COUNT[31:16]															
	W																
	R	COUNT[15:0]															
	W																

36.3.2 Detailed Register Descriptions

Table 36-4 shows the register terms.

Table 36-4. Register Terms

Term	Description
Grey bit	Unimplemented bit; always reads as zero; writing has no effect.
Access	
S	Supervisor mode only
—	Supervisor or user mode
R	
0	Always read 0.
W	
w1c	A status bit that can be read and cleared by writing a logic 1.
Reset	
0	Resets to a logic 0.
1	Resets to a logic 1.
u	Unaffected by reset.
?	Reset state is unknown.

36.3.2.1 GPT Control Register (GPTCR)

The GPT Control Register (GPTCR) is used to program and configure the GPT for appropriate operation and use of its features. IP Bus Write access to GPT Control Register (GPTCR) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while IP Bus Read access is with 0 wait state.

Address 0xBASE+0x0000 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OM3			OM2			OM1			IM2		IM1	
W	FO3	FO2	FO1													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SWR	0	0	0	0	0	FRR	CLKSRC			STOP EN	RES	WAIT EN	DB-GEN	EN-MOD	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-3. GPT Control Register (GPTCR)

Table 36-5. GPT Control Register Field Descriptions

Field	Description
31 FO3	Force Output compare channel 3. The bit causes the pin action programmed for the timer output compare 3 pin (according to the bits OM3 in this register). The OF3 flag in the status register is not affected. This bit is self negating and always read as zero. 0 Writing a 0 has no effect. 1 Causes pin action on timer output compare 3 pin, OF3 flag is not set
30 FO2	Force Output compare channel 2. The bit causes the pin action programmed for the timer output compare 2 pin (according to the bits OM3 in this register). The OF2 flag in the status register is not affected. This bit is self negating and always read as zero. 0 Writing a 0 has no effect 1 Causes pin action on timer output compare 2 pin, OF2 flag is not set
29 FO1	Force Output compare channel 1. The bit causes the pin action programmed for the timer output compare 1pin (according to the bits OM1 in this register). The OF1 flag in the status register is not affected. This bit is self negating and always read as zero. 0 Writing a 0 has no effect 1 Causes pin action on timer output compare 1pin, OF1 flag is not set
28–26 OM3	Output compare channel 3 operating Mode. This bit field specifies the response that a compare event will generate on the output pin of Output Compare channel 3. The toggle, clear and set options cause a change in the output pin only on occurrence of a compare event. When OM3 is programmed as 1xx (active low pulse), the output pin is set to one, immediately on the next input clock, and the low pulse occurs when there is a compare event. This low pulse width is input clock wide. Here input clock means clock selected by CLKSRC bits of GPT Control Register 000 Output disconnected. No response on pin 001 Toggle output pin 010 Clear output pin 011 Set output pin 1xx Generate a active low pulse of one input clock wide on output

Table 36-5. GPT Control Register Field Descriptions (continued)

Field	Description
25–23 OM2	Output compare channel 2 operating Mode. This bit field specifies the response that a compare event will generate on the output pin of Output Compare channel 2. The toggle, clear and set options cause a change in the output pin only on occurrence of a compare event. When OM2 is programmed as 1xx (active low pulse), the output pin is set to one, immediately on the next input clock, and the low pulse occurs when there is a compare event. This low pulse width is input clock wide. Here input clock means clock selected by CLKSRC bits of GPT Control Register. 000 Output disconnected. No response on pin 001 Toggle output pin 010 Clear output pin 011 Set output pin 1xx Generate a active low pulse of one input clock wide on output
22–20 OM1	Output compare channel 1 operating Mode. This bit field specifies the response that a compare event will generate on the output pin of Output Compare channel 1. The toggle, clear and set options cause a change in the output pin only on occurrence of a compare event. When OM1 is programmed as 1xx (active low pulse), the output pin is set to one, immediately on the next input clock, and the low pulse occurs when there is a compare event. This low pulse width is input clock wide. Here input clock means clock selected by CLKSRC bits of GPT Control Register. 000 Output disconnected. No response on pin 001 Toggle output pin 010 Clear output pin 011 Set output pin 1xx Generate a active low pulse of one input clock wide on output
19–18 IM2	Input capture channel 2 operating Mode. This bit field determines the transition on the input pin for Input capture channel 2 which will trigger a capture event. 00 capture disabled 01 capture on rising edge only 10 capture on falling edge only 11 capture on both edges
17–16 IM1	Input capture channel 1 operating Mode. This bit field determines the transition on the input pin for Input capture channel 1 which will trigger a capture event. 00 capture disabled 01 capture on rising edge only 10 capture on falling edge only 11 capture on both edges
15 SWR	Software reset. This is the software reset of the GPT module. It is a self clearing bit. This bit is set when the module is in reset state and is cleared when the reset procedure is over. Setting this bit resets all the registers to their default reset values except for the EN, ENMOD, STOPEN, WAITEN and DBGEN bits in this control register. 0 GPT is not in reset state 1 GPT is in reset state
14–10	Reserved bits. These are reserved bits and writing a value will not affect the functionality of GPT. These reserved bits are always read as zero. It is recommended that all writes to these bits be 0 for forward compatibility.
9 FRR	Freerun or Restart mode. This bit determines the behavior of the GPT on compare event in channel 1. In restart mode the counter resets to 0x0000_0000 and resumes counting after the occurrence of a compare event. In freerun mode, after a compare event the counter continues counting until 0xFFFF_FFFF and then rolls over to 0. 0 Restart mode 1 Freerun mode

Table 36-5. GPT Control Register Field Descriptions (continued)

Field	Description
8–6 CLKSRC	Clock Source select. These bits selects which clock will go to the prescaler and subsequently be used to run the GPT counter. This bit field value should only be changed after disabling the GPT by clearing the EN bit in this register. For other programming requirements while changing clock source, refer section 20.6.1/20-30 000 No clock 001 ipg_clk 010 ipg_clk_highfreq 011 ipp_ind_clkin (external clock from pad) 1xx ipg_clk_32k
5 STOPEN	GPT Stop Mode enable. This read/write control bit enables the operation of the GPT during stop mode. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 GPT is disabled in stop mode 1 GPT is enabled in stop mode
4 RESERVED	Reserved bits This is reserved bit and writing a value will not affect the functionality of GPT. Reading this bit will return the last written value.
3 WAITEN	GPT Wait Mode enable. This read/write control bit enables the operation of the GPT during wait mode. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 GPT is disabled in wait mode 1 GPT is enabled in wait mode
2 DBGEN	GPT debug mode enable. This read/write control bit enables the operation of the GPT during debug mode. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 GPT is disabled in debug mode 1 GPT is enabled in debug mode
1 ENMOD	GPT Enable mode. When GPT is disabled(EN=0), then both Main Counter and Prescaler Counter freeze their current count values. ENMOD bit determines the value of GPT counter when EN bit is set and Counter is enabled again. If ENMOD bit is set, then both the Main Counter and Prescaler Counter value is reset to 0 on enabling GPT again (EN=1). If ENMOD bit is programmed to 0, then both the Main Counter & Prescaler Counter restart counting from their frozen values on enabling GPT again (EN=1). If GPT is programmed to be disabled in a low power mode (STOP/WAIT), then both the main counter and the prescaler counter freeze at their current count values when GPT enters low power mode. When GPT exits the low power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit. Setting the SWR bit will clear the counter value regardless of the value of EN or ENMOD bits. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 GPT counter retain its value when it is disabled 1 GPT counter value is reset to 0 when it is disabled
0 EN	GPT Enable. This bit is the module enable bit. It is recommended that all registers be properly programmed before setting this bit. This bit is reset by a hardware reset. A software reset doesn't affect this bit. 0 GPT is disabled 1 GPT is enabled

36.3.2.2 GPT Prescaler Register (GPTPR)

The GPT Prescaler Register (GPTPR) contains bits that determine the divide value of the clock that runs the counter.

Address 0xBASE+0x0004 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	PRESCALER											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-4. GPT Prescaler Register

Table 36-6. Register Field Descriptions

Field	Description
31–12 RESERVED	Reserved bits. These are reserved bits and writing a value will not affect the functionality of GPT. These reserved bits are always read as zero.
11–0 PRESCALER	Prescaler bits. The clock selected by the CLKSRC field is divided by [PRESCALER + 1] and then used to run the counter. A change in the value of the PRESCALER bits result in Prescaler counter to reset & new count period to start immediately. See Figure 1-13.for timing diagram. 0x000 Divide by 1 0x001 Divide by 2 .... 0xFFF Divide by 4096

36.3.2.3 GPT Status Register (GPTSR)

The GPT Status Register (GPTSR) contains bits that indicate the occurrence of rollover of the counter and any event on input capture and output compare channels. The bits are write one to clear.

Address 0xBASE+0x0008 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	ROV	IF2	IF1	OF3	OF2	OF1
W											w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-5. GPT Status Register

Table 36-7. GPT Status Register (GPTSR) Field Descriptions

Field	Description
31–6	Reserved bits. These are reserved bits and writing a value will not affect the functionality of GPT. These reserved bits are always read as zero.
5 ROV	Rollover Flag. This bit indicates that the counter has reached its maximum possible value and rolled over to 0 from which it continues counting. This bit is only set if the counter has reached 0xffffffff in both restart and freerun modes. 0 Rollover not occurred 1 Rollover occurred
4 IF2	Input capture 2 Flag. This bit indicates that a capture event has occurred on Input Capture channel 2. 0 Capture event not occurred 1 Capture event occurred
3 IF1	Input capture 1 Flag. This bit indicates that a capture event has occurred on Input Capture channel 1. 0 Capture event not occurred 1 Capture event occurred
2 OF3	Output Compare 3 Flag. This bit indicates that a compare event has occurred on Output Compare channel 3. 0 Compare event not occurred 1 Compare event occurred
1 OF2	Output Compare 2 Flag. This bit indicates that a compare event has occurred on Output Compare channel 2. 0 Compare event not occurred 1 Compare event occurred
0 OF1	Output Compare 1 Flag. This bit indicates that a compare event has occurred on Output Compare channel 1. 0 Compare event not occurred 1 Compare event occurred

36.3.2.4 GPT Interrupt Register (GPTIR)

The GPT Interrupt Register (GPTIR) contains bits that control the generation of interrupt on rollover, input capture and output compare events.

Address 0xBASE+0x000C Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	ROVIE	IF2IE	IF1IE	OF3IE	OF2IE	OF1IE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-6. GPT Interrupt Register (GPTIR)

Table 36-8. GPT Interrupt Register Field Descriptions

Field	Description
31–6 RESERVED	Reserved bits. These are reserved bits and writing a value will not affect the functionality of GPT. These reserved bits are always read as zero.
5 ROVIE	Rollover Interrupt Enable. This bit controls the occurrence of rollover interrupt. 0 Interrupt disabled 1 Interrupt enabled
4 IF2IE	Input capture 2 Interrupt Enable. This bit controls the occurrence of input capture channel 2 interrupt. 0 Interrupt disabled 1 Interrupt enabled
3 IF1IE	Input capture 1 Interrupt Enable. This bit controls the occurrence of input capture channel 1 interrupt. 0 Interrupt disabled 1 Interrupt enabled
2 OF3IE	Output Compare 3 Interrupt Enable. This bit controls the occurrence of output compare channel 3 interrupt. 0 Interrupt disabled 1 Interrupt enabled
1 OF2IE	Output Compare 2 Interrupt Enable. This bit controls the occurrence of output compare channel 2 interrupt. 0 Interrupt disabled 1 Interrupt enabled
0 OF1IE	Output Compare 1 Interrupt Enable. This bit controls the occurrence of output compare channel 1 interrupt. 0 Interrupt disabled 1 Interrupt enabled

36.3.2.5 GPT Output Compare Register 1 (GPTOCR1)

The GPT Compare Register 1 (GPTOCR1) holds the value that determines when a compare event will be generated on output compare channel 1. Any write access to the compare register of channel 1 while in restart mode (FRR=0) will result in the GPT counter being reset.

IP Bus Write access to GPT Output Compare Register1 (GPTOCR1) results in one cycle of wait state (ips_xfr_wait high for 1 cycle), while IP Bus Read access is with 0 wait state.

Address 0xBASE+0x0010 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMP[31:16]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 36-7. GPT Output Compare Register 1 (GPTOCR1)

Table 36-9. GPT Output Compare Register 1 Field Descriptions

Field	Description
31–0 COMP	Compare Value. When the counter value equals this bit field value a compare event is generated on output compare channel 1.

36.3.2.6 GPT Output Compare Register 2 (GPTOCR2)

The GPT Compare Register 2 (GPTOCR2) holds the value that determines when a compare event will be generated on output compare channel 2.

Address 0xBASE+0x0014 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMP[31:16]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 36-8. GPT Output Compare Register 2 (GPTOCR2)

Table 36-10. GPT Output Compare Register 2 Field Descriptions

Field	Description
31–0 COMP	Compare Value. When the counter value equals this bit field value a compare event is generated on output compare channel 2.

36.3.2.7 GPT Output Compare Register 3 (GPTOCR3)

The GPT Compare Register 3 (GPTOCR3) holds the value that determines when a compare event will be generated on output compare channel 3.

Address 0xBASE+0x0018 Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	COMP[31:16]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	COMP[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 36-9. GPT Output Compare Register 3 (GPTOCR3)

Table 36-11. GPT Output Compare Register 3 Field Description

Field	Description
31–0 COMP	Compare Value. When the counter value equals this bit field value a compare event is generated on output compare channel 2.

36.3.2.8 GPT Input Capture Register 1 (GPTICR1)

The GPT Input Capture Register 1 (GPTICR1) is a read-only register which holds the value that was in the counter during the last capture event on input capture channel 1.

Address 0xBASE_001C Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CAPT[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-10. GPT Input Capture Register 1 (GPTICR1)

Table 36-12. GPT Input Capture Register 1 Field Descriptions

Field	Description
31–0 CAPT	Capture Value. On occurrence of a capture event on Input capture channel 1, the current value of the counter is loaded into this register.

36.3.2.9 GPT Input Capture Register 2 (GPTICR2)

The GPT Input Capture Register 2 (GPTICR2) is a read-only register which holds the value that was in the counter during the last capture event on input capture channel 2.

Address 0xBASE+0x0020 Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CAPT[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CAPT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-11. GPT Input Capture Register 2 (GPTICR2)

Table 36-13. Register Field Descriptions

Field	Description
31–0 CAPT	Capture Value. On occurrence of a capture event on Input capture channel 1, the current value of the counter is loaded into this register.

36.3.2.10 GPT Counter Register (GPTCNT)

The GPT Counter Register (GPTCNT) is the main counter register. It is a read-only register and can be read without affect the counting process of the GPT.

Address 0xBASE+0x0024 Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNT[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 36-12. GPT Counter Register (GPTCNT)

Table 36-14. GPT Counter Register Field Description

Field	Description
31–0 COUNT	Counter Value. These bits show the current count value in the counter register.

36.4 Functional Description

The General Purpose Timer (GPT) has a single counter (GPTCNT). It is a 32-bit free-running up counter which starts counting after it is enabled by software (EN = 1). The counters clock source is the output of the prescaler labelled “Prescaler output” in Figure 36-13. If the GPT is programmed to be disabled in a low power mode (STOP/WAIT), then both the main counter and the prescaler counter freeze at their current count values when GPT enters low power mode. When GPT exits the low power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit. GPTCNT can be read at any time by the processor. Both input capture channels use the same counter (GPTCNT).

A hardware reset resets all the GPT registers to their respective reset values. All registers except the output compare registers (GPTOCR1,GPTOCR2, and GPTOCR3) obtain a value of 0x0. The compare registers are reset to 0xFFFF_FFFF. There is a software reset available (SWR bit in control register) which can reset all the register bits except the EN, ENMOD, STOPEN, WAITEN and DBGEN bits. The state of these bits are not affected by a software reset. Software reset can be given even when the GPT is disabled.

36.4.1 Clocks

The clock that feeds the prescaler can be selected from the following clock inputs

- **High frequency Clock** (ipg_clk_highfreq)

  This is a high frequency clock which is provided by the Clock Controller Module (CCM). This clock is supposed to be on in low power mode when the ipg_clk is turned off. Thus the GPT can be run on this clock in low power mode. The CCM is expected to provide this clock after synchronizing it to ahb_clk in normal functional mode and switch to the unsynchronized version in the low power mode.

- **Low Reference Clock** (ipg_clk_32k)

  This is the 32 kHz low reference clock which is provided by the CCM. This clock is supposed to be on in low power mode when the ipg_clk is turned off. Thus the GPT can be run on this clock in low power mode. The CCM is expected to provide this clock after synchronizing it to ahb_clk in normal functional mode and switch to the unsynchronized version in the low power mode.

- **External Clock** (ipp_ind_clkin)

  This is the external clock from outside the chip which can be used to run the counter. This clock is treated as asynchronous to ipg_clk and synchronized to ipg_clk inside the module. Thus its frequency is limited to < 1/4 frequency of ipg_clk for proper functioning of the GPT. Also in low power modes if ipg_clk is not available it cannot be used to run the counter.

- **Global Functional Clock** (ipg_clk)

  This clock is supposed to be on in normal operations, if ipg_clk or ipp_ind_clkin is selected(CLKSRC=001 or 011) as Clock Source. In low power modes, if GPT is programmed to be disabled (STOPEN or WAITEN or DOZEN=0), then ipg_clk can be switched off.

  The clock input source is determined by the clock source (CLKSRC) field in the control register. The clock input to the prescaler can also be disabled by programming the CLKSRC bits of control register to 000. **This field (CLKSRC) value should be changed only after disabling the GPT by setting the EN bit in the GPTCR to 0.** For other programming requirements when changing clock source, see Section 36.5, Initialization/Application Information.”

  The PRESCALER field is used to select the divide ratio of the input clock that drives the main counter. The prescaler can divide the input clock by a value from 1 to 4096 and can be changed any time. A change in the value of the PRESCALER field is immediately reflected on its output clock frequency.

Figure 36-13 shows the prescalar value change diagram.

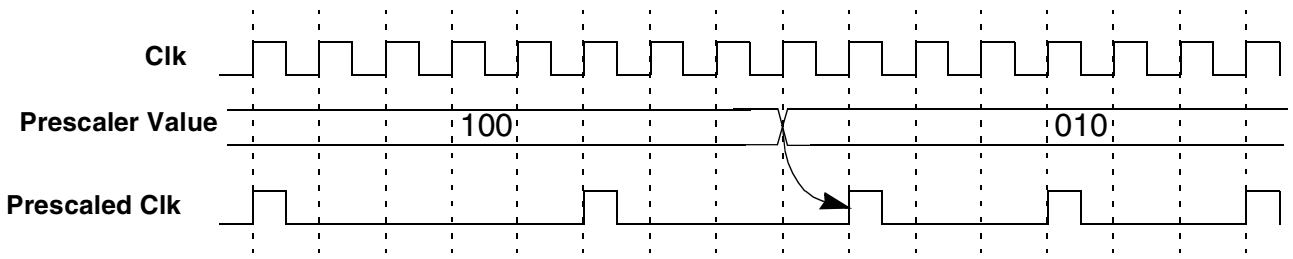
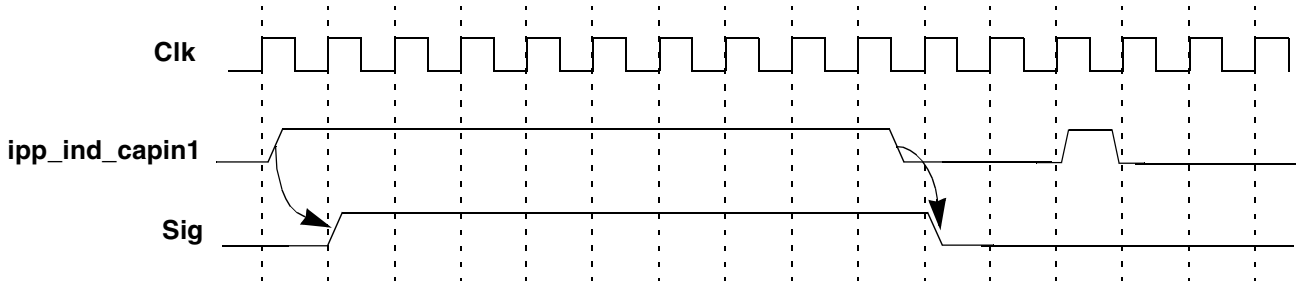


Figure 36-13. Prescaler value change diagram

36.4.2 Input Capture

There are two input capture channels and each input capture channel has a dedicated capture pin, capture register and input edge detection/selection logic. Each input capture function has an associated status flag, and can cause the processor to make an interrupt service request. When a selected edge transition occurs on an input capture pin, the contents of the GPTCNT is captured on the corresponding capture register and sets the appropriate interrupt status flag. An interrupt request can be generated when the transition is detected if its corresponding enable bit is set in the Interrupt Register. The capture can be programmed to occur on the input pin rising edge, falling edge, on both edges or can be disabled completely. The events are synchronized with the clock selected to run the counter. Only those transitions which occur at least a single clock cycle (clock selected to run the counter) after the previous recorded transition will be guaranteed to trigger a capture event. There can be up to one clock cycle of uncertainty in latching of the input transition. The input capture registers can be read at any time without affecting their values.

Figure 36-14 shows the input capture event timing diagram.



Clk - The clock selected from CLKSRC bit field setting
ipp_ind_capin1 - Pad signal for capture channel 1
Sig - Capture signal as sensed by the module

Figure 36-14. Input Capture Event Timing Diagram

36.4.3 Output Compare

The three output compare channels use the same counter (GPTCNT) as the input capture channels. When the programmed content of an output compare register matches the value in GPTCNT, an output compare status flag is set and an interrupt is generated if the corresponding bit is set in the interrupt register. Consequently, the output compare timer pin will be set, cleared, toggled, not affected at all or provide an active-low pulse for one input clock period (this is subject to the restriction on the maximum frequency allowed on the pad) according to the mode bits programmed. There also exists a forced-compare feature allowing the software to generate compare event when required without the condition of the counter value being equal to the compare value. The action taken as a result of a forced compare is the same as when an output compare match occurs, except that the status flags are not set and no interrupt can be generated. Forced channels take programmed action immediately after the write to the force-compare bits. These bits are self-negating and always read as zeros.

Figure 36-15 shows the output compare and interrupt timing diagram.

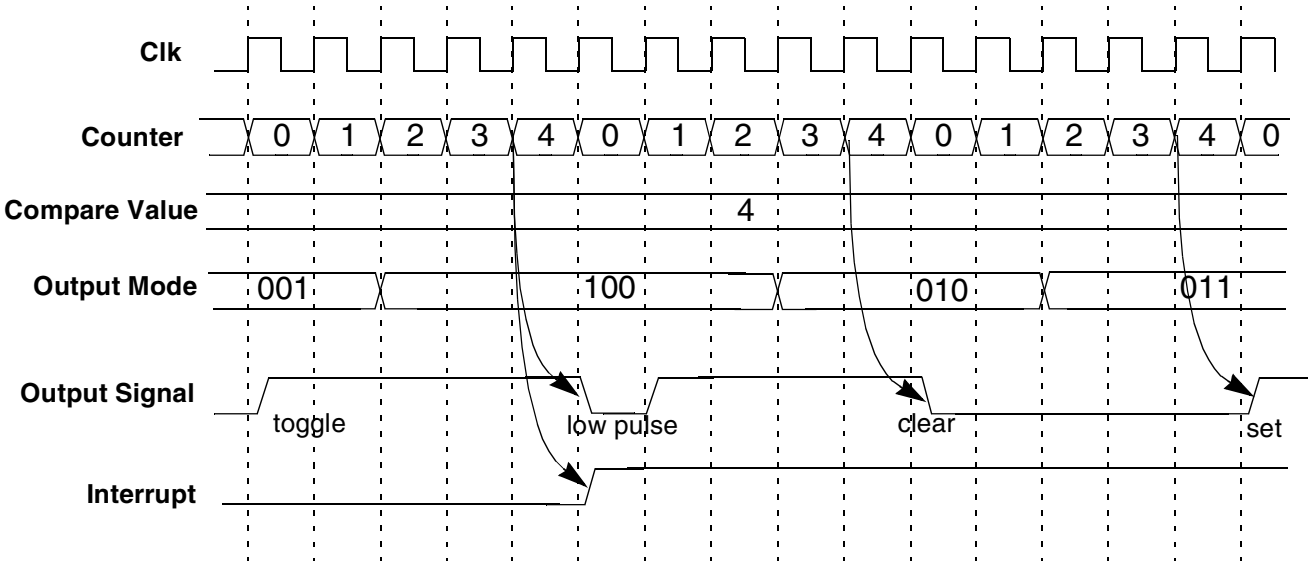


Figure 36-15. Output Compare and Interrupt Timing Diagram

36.4.4 Interrupts

There are six different interrupts that are generated by the GPT. All interrupts can be generated in low power and debug modes if the selected clock for running the counter is available.

- Rollover Interrupt

  This interrupt is generated when the GPT counter reaches 0xFFFF_FFFF and resets to 0x0000_0000 and continues counting. The interrupt is enabled by the ROVIE bit in GPTIR register and its associated status bit is ROV bit in GPTSR register.

- Input Capture Interrupt 1 and 2

  On occurrence of a capture event, interrupts can be generated by each input capture channel. These interrupts are enabled by IF2IE and IF1IE bits in GPTIR register and their associated status bits are IF2 and IF1 in GPTSR register. The capture of the counter value due to a capture event is not affected by a pending capture interrupt. Capture register is updated with new counter value on occurrence of capture event irrespective of that capture channels interrupt has been serviced or not.

- Output Compare Interrupt 1, 2, and 3

  On occurrence of a compare event, interrupts can be generated by each output compare channel. These interrupts are enabled by OF3IE,OF2IE and OF1IE bits in GPTIR register and their associated status bits are OF3,OF2 and OF1 in GPTSR register. No interrupt is generated due to a forced compare.

  An cumulative interrupt line is also present which is asserted whenever any of the above interrupts are posted. This has no associated enables or status bits.

36.4.5 Low-Power Mode (LPM) Behavior

In low power modes if the clock from the selected clock source is available (except ipp_ind_clkin which can be used only if ipg_clk is available), the counter continues to run depending on whether the control bit for that mode is set. In absence of the clock itself or the corresponding low power bit in control register being 0, the main counter and the prescaler counter freeze at their current values and resume counting from their frozen values when the low power mode is exited.

36.4.6 Debug Mode Behavior

In debug mode, the modules have the option of continuing to run or be halted. If the DBGEN bit is not set in the GPTCR, the GPT timer is halted. If the DBGEN bit is set, then the GPT timer will continue to run in debug mode.

36.5 Initialization/Application Information

This section describes how to change the clock source.

The CLKSRC field in GPTCR determines the clock source. This field value should be changed only after GPT has been disabled (EN=0). Use the following software sequence when changing the clock source.

1. Disable GPT by setting EN=0 in GPTCR.
2. Disable GPT interrupt register (GPTIR).
3. Program OM3,OM2,OM1 to 00 in GPTCR.
4. Change clock source CLKSRC to desired value in GPTCR.
5. Clear GPT status register (GPTSR) i.e. (w1c).
6. Enable GPT interrupt register (GPTIR).
7. Set ENMOD=1 in GPTCR, to bring GPT counter to 0x00000000.
8. ENABLE GPT (EN=1) in GPTCR.

Chapter 37
Graphics Processing Unit 2D (GPU2D)

37.1 Overview

The GPU is an embedded 2D and vector graphics accelerator targeting the OpenVG 1.1 graphics API and feature set. It accelerates 2D bitmap graphics operations, such as BitBlt, fill and raster operations, using a separate 2D graphics acceleration unit. Vector graphics rendering is accelerated by a separate anti-aliasing polygon rasterizer, which is connected to the 2D graphics acceleration unit. The core has a rich and well-chosen set of features that emphasize very high image quality and low memory bandwidth consumption.

The GPU top level block diagram is presented in Section 37.3, GPU2D Block Diagram."

37.2 GPU Feature List

The following sections describe the functional features of the graphics processor.

37.2.1 Frame Buffer

Frame buffer features are as follows:

- Frame buffer sizes supported up to 2048 × 2048
- ARGB4444, RGB565, ARGB1555, ARGB5551, ARGB8888 frame buffer modes
- Configurable ARGB order in frame buffer: ARGB, BGRA, ABGR, RGBA
- Linear and block-based (4 × 4 pixels) frame buffer modes
- Fast buffer clears
- Support for OpenVG render to Image

37.2.2 2D Bitmap Graphics (Separate 2D Unit)

The 2D bitmap graphics features are as follows:

- Parallel operation with the 3D pipeline, independent command input
- BitBlt (surface-to-surface copy)
- Format conversion from monochrome/ARGB/YUV to ARGB during BitBlt
- Block fill
- Internal 32-bit color precision

The source bitmap format is as follows:

- 1/4/8-bit monochrome
- ARGB4444, RGB565, ARGB1555, ARGB5551, ARGB8888
- Configurable ARGB order: ARGB, BGRA, ABGR, RGBA
- Packed YUV 4:2:2 formats (FOURCC codes YUY2, UYVY, YVYU), two pixels per 32 bits of data
- 1-bit bitmap maps to foreground and background colors
- 4-bit bitmap is optionally gamma corrected to 8-bit alpha values and can be combined with foreground color to draw anti-aliased fonts

The destination bitmap format is as follows:

- ARGB4444, RGB565, ARGB1555, ARGB5551, ARGB8888, B8, A8, AB88
- Configurable ARGB order: ARGB, BGRA, ABGR, RGBA
- Supports three source bitmaps for separate mask/pattern/alpha bitmap support plus reading destination for ROP, blend and color key operations
- Supports masking source coordinates for wrapping patterns
- Supports ROP4 (ROP3 with separate ROPs for masked and unmasked pixels) logical operations
- Supports inverting mask and alpha values from source
- Supports destination rotation by 0/90/180/270 degrees
- Supports programmable blending with optional alpha un-premultiply
- Supports per pixel and constant alpha with optional modulation by source color alpha for OpenVG alpha masking
- Supports color keying by source and destination colors, with optional ignoring of alpha channel
- Supports one scissor rectangle for destination coordinates
- Dithering (ordered)
- Color component masking
- sRGB reads and writes
- Non-power of two source and destination bitmap sizes supported (stride must be a multiple of 32-bits)
- BitBlt with scaling implemented with the 3D rendering pipeline, bilinear filtering with texture lookups, programmable filter kernels possible with the programmable Pixel processor

Vector Graphics

Vector graphic features are as follows:

- Parallel operation with the 3D pipeline, independent command input
- Rasterization of convex and concave polygons with anti-aliasing
- Efficient native polygon rendering (no tessellation to triangles)
- Non-zero and odd-even fill rules
- Primitives supported:

- Polygons
- OpenVG path primitives (except Elliptical Arcs): Horizontal/vertical lines, generic lines, curves, smooth curves, move to, path closing
- Curve types supported: cubic and quadratic BÈzier
- Strokes with thickness, joints and end caps, unlimited stroke thickness
- Special case handling of singularities for thick strokes
- Supports paths with a maximum of 256 crossings along a horizontal or a vertical line

- Input coordinates
  - Absolute and relative coordinate input in floating point
  - Fixed-point (byte, short, int) and floating-point coordinate input - 0.8, 0.16, 16.16 formats
  - Little- and Big-endian support separately selectable for command stream and data.
- Geometry
  - User to surface transform for vertices and stroke shape
  - Hardware curve tessellation
  - Adjustable accuracy for curve and round cap splitting
  - OpenVG/SVG join types: Miter (with miter limit), round, bevel
  - OpenVG/SVG cap types: Butt, round, square
- Pixel processing
  - Programmable gradient and texturing processor
  - Linear and radial gradients (with focal point)
  - Perspective texture mapping with filtering
  - Two textures supported
  - sRGB and pre-multiply support for textures
  - 16-sample anti-aliasing
  - 4x RGSS AA (Rotated Grid Super Sample)
  - Per-pixel alpha-masking
  - Maximum texture size: 1024x1024 pixels
- Vector graphics rendering system CPU load:
- Display list generation during path creation ñ commands and vertices are stored to an internal format/buffer, no format conversion is performed
- Filling or stroking a path only requires a few register writes to start the operation in hardware
- Display lists are transferred to the vector graphics rasterizer using DMA without CPU interaction

37.3 GPU2D Block Diagram

Figure 37-1 shows the GPU top-level block diagram.

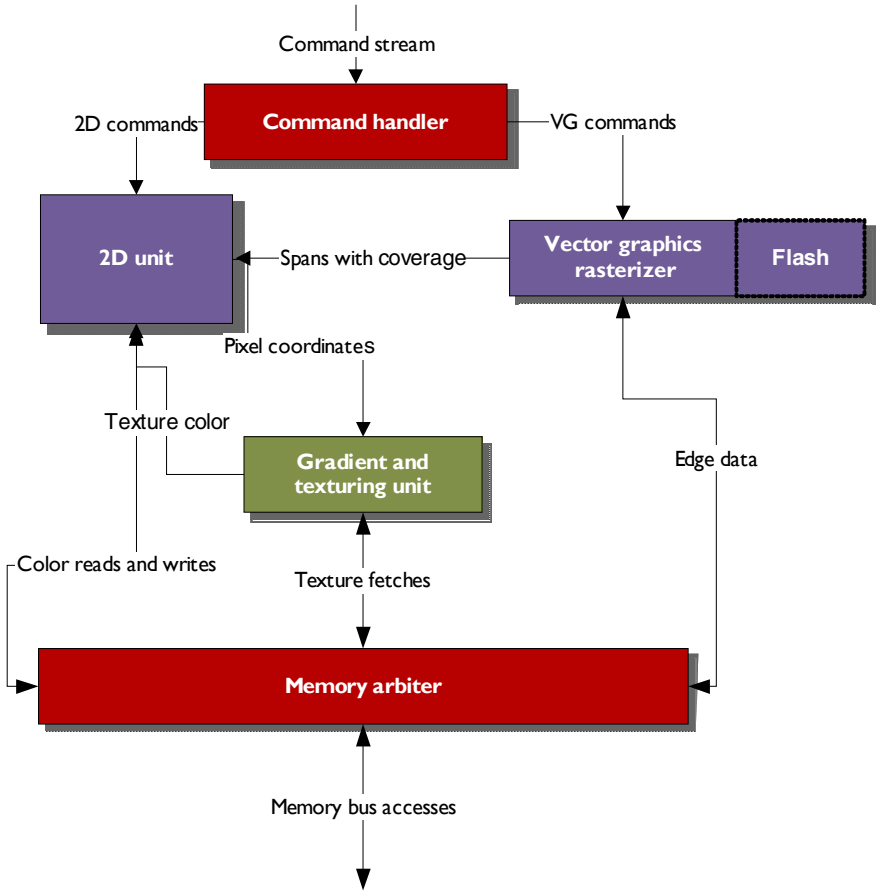


Figure 37-1. GPU Top-Level Block Diagram

37.4 Modes of Operation

The GPU supports the following:

- 2D bitmap acceleration mode
- Vector Graphic acceleration mode
- Low-power mode

37.5 Reset

The GPU2D uses one reset port, which is named HRESET*n*. The internal Z160 core (G12) contains asynchronous reset signals for each corresponding clock domain that is directly connected to HRESET*n*, namely rst_n for the clk clocks and bus_rst_n for the busclk clock.

Resets are active low. The reset signals asynchronously reset all of the DFFs in the design.

37.6 Interrupts

The Z160 Core (G12) generates individual interrupts internally. Each interrupt can be enabled or disabled by changing its own enable bit. Setting the enable bit high enables the corresponding interrupt.

The interrupts from all sources in the Z160 Core (G12) are combined (ORed) and output as the active low **gpu2d_int_b** interrupt.

37.7 DMA

GPU2D is a DMA master. It reads commands and writes data to system memory through EMI. GPU2D can also be fed by DMA through MAX slave port 0.

37.8 Memory Map

The Z160 Graphics Core (G12) memory map is described in the following sections:

- AHB slave interface
- AXI master memory interface (EMI port)

37.8.1 AHB Slave Interface

Table 37-1 shows the memory map for the AHB Slave Interface port of the GPU. This is based on the GPU occupying slave port 0 of the crossbar, whose base address is 0x2000_0000. GPU2D has two kinds of registers: interface registers, which can be read/write accessible, and internal registers, which can only be written through Z160's slave ports.

Table 37-1. GPU2D memory map

Description	Memory Address Base	Memory Address End
Registers	0xD000_0000	0xD000_07FC

A description of all user-accessible registers in the design can be found from regs.html.

Interface registers can only be accessed via the status read/write channel directly through the slave port, which is mapped to 0x400–0x7FC address range. These registers are used to read and clear interrupts by the CPU.

Internal 2/VG registers are write only. They can only be accessed via two writing commands through the slave port (GPU2D's base address). The first command is register address write, and the second command is flowing a data write. This means that the command stream, which is prepared by the software driver, is written through the 0x000–0x3ff address range either directly or through DMA operation by the core. These registers are not accessible directly by slave port channels just as the interface registers are not accessible through this channel.

37.8.2 AXI Master Memory Interface (EMI port)

The Z160 Graphics Core (G12 or GPU2D) can access memory with or without a memory management unit (MMU).

Translation is performed using a table with 8-Kbyte entries, one for each 4-Kbyte page in a 32-Mbyte linear address space. For further information, please refer to “Yamato_MMU” released by AMD.

Chapter 38
3D Graphics Accelerator (GPU3D)

38.1 Overview

The GPU3D (3D graphics processing unit) is based on the AMD Z430 (also known as ATI Yamato DX). The contains an embedded engine capable of DirectX9 Shader Model 3.0+ program execution. The module focus is on accelerating user level graphics APIs, such as OpenGL ES 2.0 and 1.1, and Direct3D Mobile 1.2.

38.1.1 IGPU3D Features

The GPU3D has the following high-level features:

- Built to accelerate OpenGL ES 2.0 and Direct3D Mobile 1.2
- Unified Shader Architecture
  - Uses dynamically shared shader ALU/memory resources between vertex and pixel shaders.
- General purpose exports to system memory from Vertex & Pixel Shaders
- Supports 2- and 4- sample MSAA
- Command Processor:
  - As a DMA master, support advanced packet based command stream manager allowing for efficient and flexible transfer of graphics commands and host data from the host system to the graphics processor core
- Graphics Memory Controller and Graphics Memory (GMEM)
  - Customer configurable on-chip memory used to accelerate 3D rendering using a binning architecture significantly reducing external system bandwidth requirements. The i.MX51 uses a 128 Kbyte internal GMEM.
- Integrated Power Management
  - Block-level clock gating managed automatically in the IP.

38.2 Capabilities and Performance

Its capabilities and performance features are as follows:

- 32-bit FP internal shader precision
- Supports general purpose exports to system memory from vertex and pixel shaders.
- Uses indexed vertex data fetches from the vertex shader as a high-bandwidth vertex data path.

- Latency hiding via FIFOs and multi-threading
- Sophisticated shader support
  - 512 4-component constants
  - 1024 shader instructions
  - 16 textures
  - 16 4-component interpolants
  - 64 general-purpose registers
- Rich texture format and types
  - Compressed
  - 16-bit Floating point
  - Cube map
  - Volume textures
  - Non power-of-2
  - Anisotropic
- 1 primitive (triangle/line/point) every 6 clocks and 1 vertex every 6 clocks the lower rate element limited by the ratio
- 2 vector (4 component) and 1 scalar (single component) ALU instruction per clock
- 1 control flow instructions (jumps, loops) per clock
- 1 export per clock
- 14 component pixel shader input interpolant per clock
- Early-Z testing at up to 4 pixels per clock
- 1 pixel (at 4 sample MSAA) per clock with alpha blending and depth test

38.3 GPU3D Block Diagram

The GPU3D contains three primary blocks:

- Graphics Core (GC)
  - Programmable graphics engine that performs all of the data processing and memory transactions
- Graphics Arbiter (GARB)
  - Controls the GMEM and arbitrates between requests from the GC and the system
- Graphics Memory (GMEM)
  - Local SRAM buffers (4x32KB buffers) that are used for the tile/bin of pixels when the GC is working.

Figure 38-1 shows the GPU3D top-level block diagram.

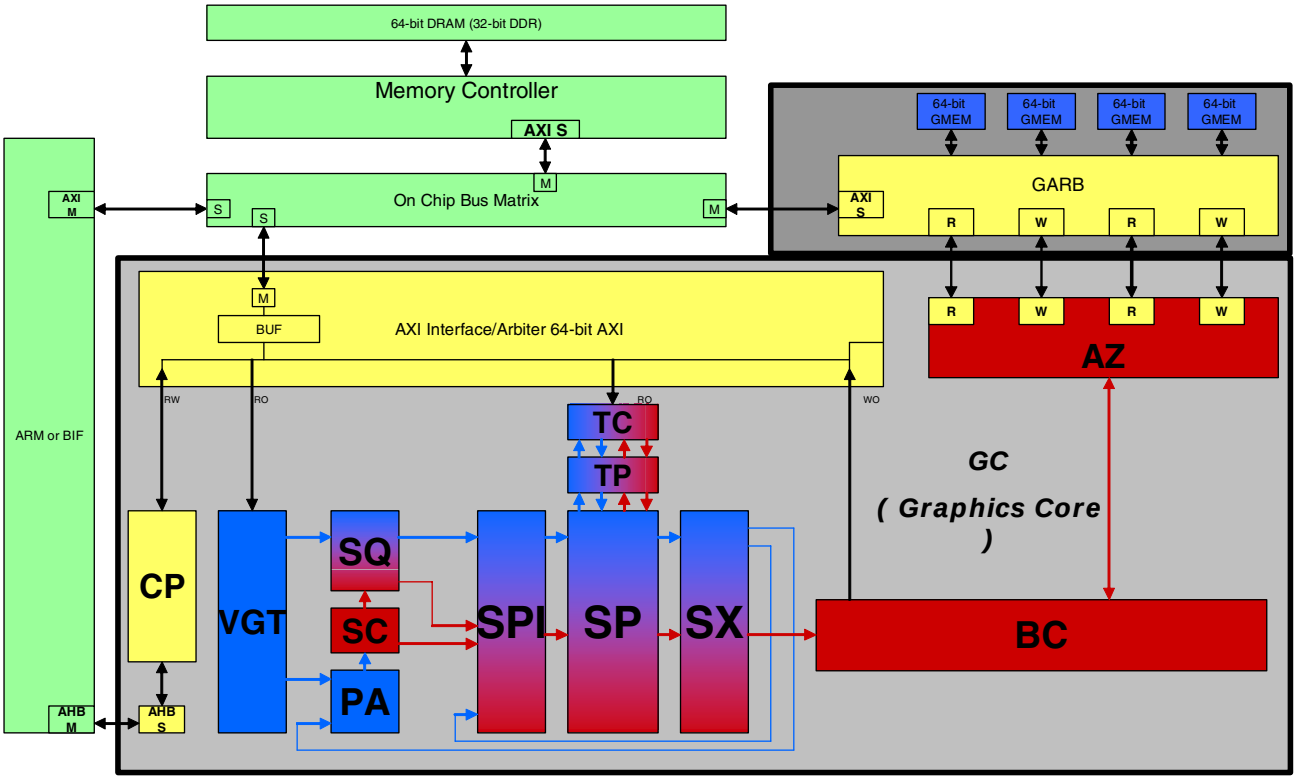


Figure 38-1. GPU3D Top-Level Block Diagram

38.4 GPU3D SoC Interface

38.4.1 GPU3D SoC Integration Top Level Diagram

Figure 38-2 shows the GPU3D graphics core connection in the i.MX51 system.

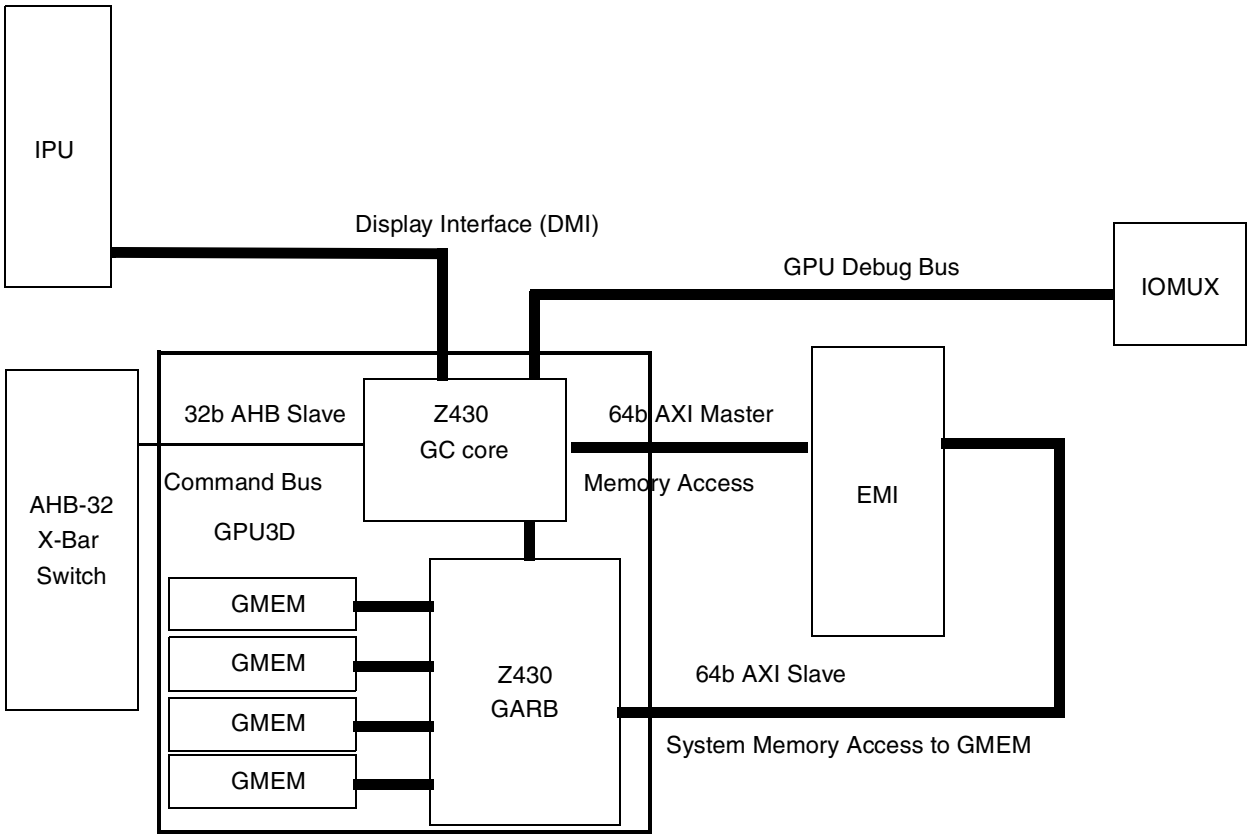


Figure 38-2. GPU3D System Connectivity

38.4.2 SoC Interface Summary

Key features of the SOC interface are as follows:

- Slave Interface to CPU or controller interface
  - — Peripheral AMBA 2.0 AHB-Lite Host Interface
  - — Provides host access to graphics core from the system processor.
  - — 32b wide, synchronous with core.
  - — Support for clock gating for phase removal on AHB clock.
- AXI Master Interface into memory system (AXI V1.0)
  - — 64b wide, synchronous
  - — Provides access to system memory from the graphics core.
  - — Programmable number of read and write requests to the memory system

- Number of outstanding request limited by system resources.

- AXI slave interface for system access to graphics memory (though GARB to GMEM)
  - 64b wide, synchronous interface
  - 1 outstanding read and write request
- Display Module Interface (DMI)
  - For synchronization with the display processor (IPU)
  - Provides indication to display processor from graphics core that a newly rendered frame is ready and that the display must perform a buffer flip at the next vertical blank.
  - Provides indication to graphics core from the display processor that the buffer identified by buffer ID has been displayed at least once..
- Control Interface Block (CIB on AHB slave bus)
  - Holding block for clock gating, reset etc.
  - Interrupt Interface (INT)
  - Error and status interrupts to the controller.
- Design For Test Interface (DFT)
  - Memory test collar access.
  - Scan/BIST access.
- Debug bus
  - GPIO register as control functions
  - Set of signals OR chained between internal masters for debug purposes
  - 32b bus muxed to 16b output to pad

38.4.3 Memory Interface Detail

This section provides a more detailed explanation of the memory interface.

38.4.3.1 Access type

The GPU3D core generates 128 and 256 bit bursts to external memory from its internal masters, which are arbitrated by a Memory Hub (MH). The MH supports up to 64 outstanding transactions, 8 AXI IDs, and contains a MMU.

Typical graphics use cases (usually games) will utilize approximately 400 to 500 MB/s and require a memory latency of less that 90ns (70ns is preferred).

The Memory Hub selects between the following clients in the graphics core:

- CP—(Command Processor) Read/Write
- VGT—(Vertex Grouper and Tesselator) Read Only
- TC—(Texture Cache) Read only
- RB—(Render Block) Write Only

Sub-Client	Sub-Client Designation	AXI ID	Operation	Transaction Size (Bytes)	Sub-Client Outstanding Transaction Limit
Ring Buffer	CPr0	0	Read	32	Programmable (Note 1)
Indirect Buffer #1	CPr1	1	Read	32	Programmable (Note 1)
Indirect Buffer #2	CPr2	2	Read	32	Programmable (Note 1)
State Sub-Block, Constant, & Shader Instruction Data	CPr3	3	Read	32	Programmable (Note 1)
Micro-Engine & Write Pointer Polling	CPr4	4	Read	32	1 Micro-engine read & 1 write pointer read
VGT Indices	VGTr0	5	Read	32	5 (Note 2)
VGT Bin ID	VGTr1	6	Read	32	8
Texture/Vertex Read	TCr	7	Read	16/32	9
Synchronization Semaphores, Micro-Engine Semaphores, Constant State Data	CPw	Programmable	Write	16/32	8 Unconfirmed writes, unlimited writes past point of confirmation
RB Copy	RBw	Programmable	Write	32	No client limit
PA	PAw	Programmable	Write	32	1
MMU TLB Miss	MMUr	Programmable	Read	32	1

All transactions are incrementing address bursts, as follows.

- The starting address is always aligned to the burst size, meaning that all 16-Byte bursts start at a 16-Byte aligned address, and all 32 Byte bursts start at a 32-Byte aligned address.
- Command stream and vertex data is read in bursts of 256 bits.
- The burst cache combines color accesses from several smaller objects to 4×4 pixel (16 bit), or 4×2 pixel (3 bit), which are transferred to memory in bursts of 256 bits.
- Color writes use byte enables when incomplete pixel blocks are written to the memory

38.4.3.2 Memory Management

The GPU3D GC core includes a memory management unit (MMU) capable of remapping a programmably sized region of the 32 bit GC memory space. This unit uses a single-level mapping table, with each 32 bit table entry mapping a 4 Kbyte region. Hence to map a 1 Mbyte region requires 256 table entries or a table 1K in size.

To make the remapping more efficient, the GC includes a translation lookaside buffer (TLB), which acts as a cache of translations. This cache holds a total of 128 translations organized as 16 lines of 8 translations each. Each of the 16 lines is fully associative, allowing translations for sections of up to 16 different surfaces to be simultaneously stored in the TLB.

Each client within the GC can be selected whether or not to use the MMU remapping function. Any accesses that do not use the MMU may be limited to a programmable range of physical address space, with accesses outside of that region resulting in a page fault fatal error. For more information, please refer to AMD's "ATI Yamato MMU."

When the driver requests memory from the OS, the virtual to physical translation is stored in the GPU3D's page table. The page table is a contiguous physical chunk of memory whose starting address is defined as the page table base (PTB) and whose extent depends on the amount of virtual memory required by the

driver. For example, if supporting 4-KByte pages and each page table entry is 32b, a virtual address range of 64 MB available to the driver would require 16K entries or 64KB of page table storage.

Note that the page table used by the graphics is not the same as the page table used by the ARM processor. Specifically, the GPU page table is a single level page table, and not all access control mechanisms are implemented.

In its simplest form, using a 4-KByte page, bits [11–0] of the virtual address can be the same as the physical address[11–0] and can be used as an index into the page. Bits [31–12] of the virtual address, in conjunction with the Virtual Address Base register, can be used as an index into the page table to find the address of the relevant page table entry (PTE).

Figure 38-3 and Figure 38-4 show the basic concept of the GPU3D MMU page table.

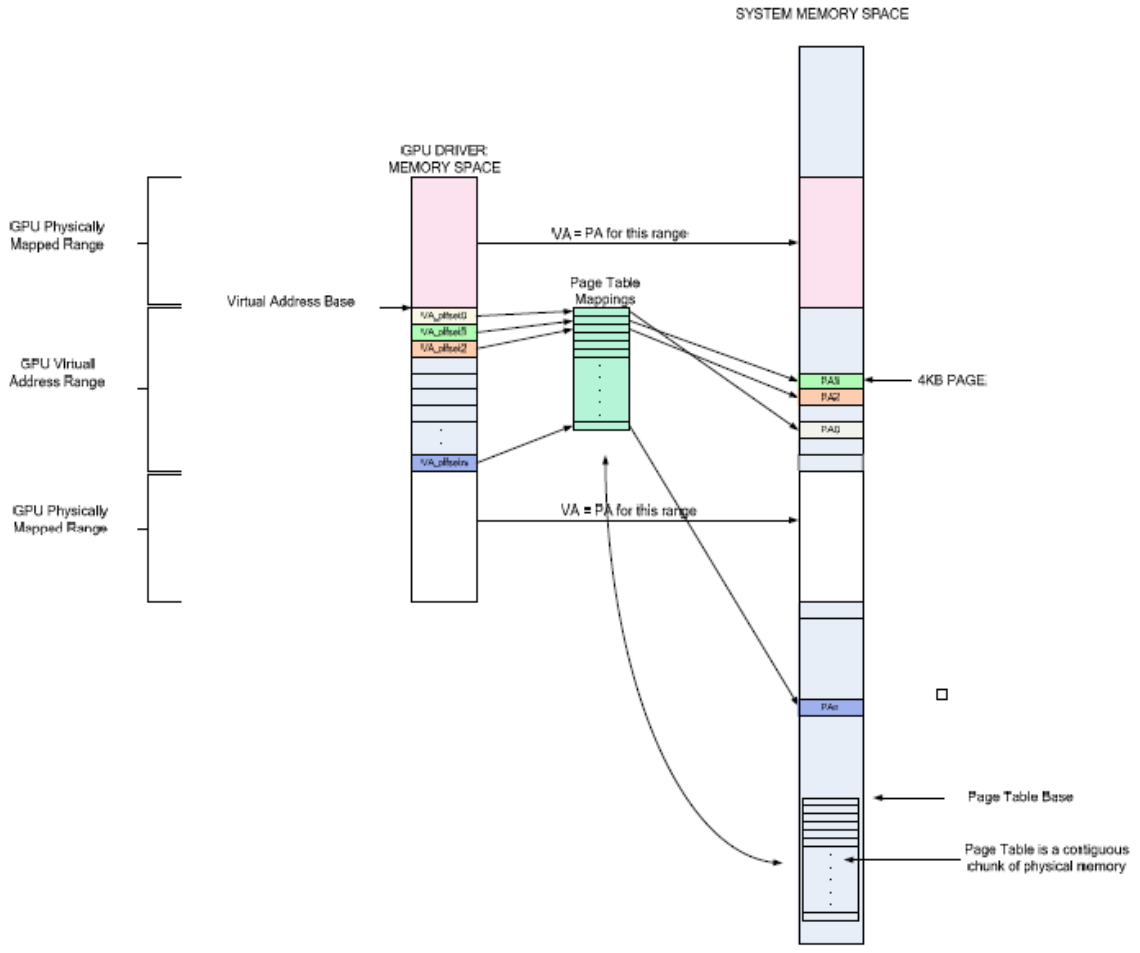


Figure 38-3. GPU3D Memory Mapping Concept

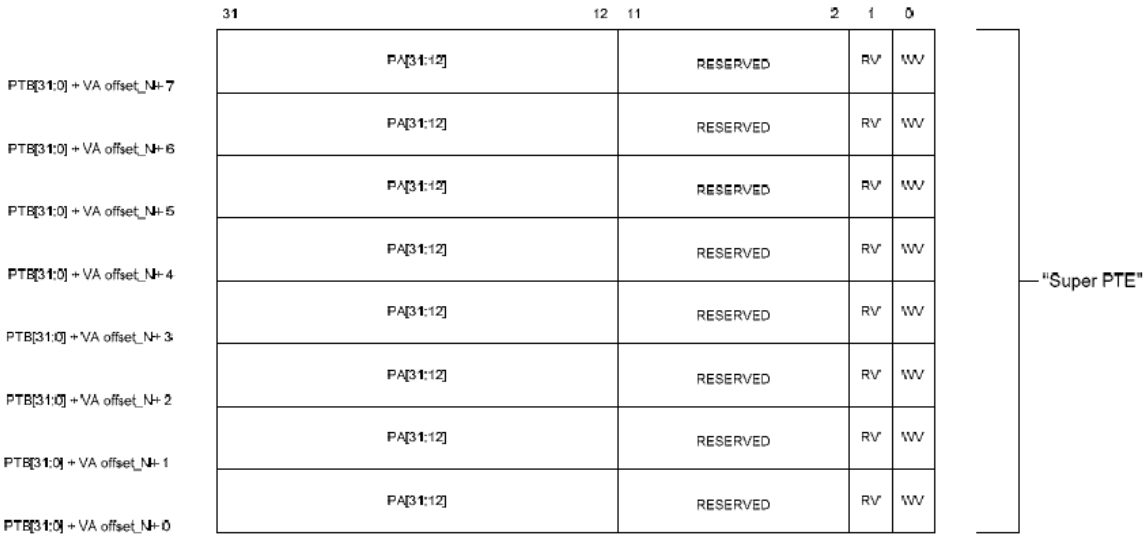


Figure 38-4. Page Table Entry Format in External Memory

38.4.4 DMI Interface Detail

Table 38-1 list the DMI handshake ports between GPU3D and IPU.

Table 38-1. GPU3D DMI Ports

Name	Type	Source	Description
gpu_use_bufid[2:0]	Output	GPU3D	Tell IPU which buffer can be displayed, one-hot coded
ipu_vsync_valid	Input	IPU	Tell GPU3D one frame is finished, high in vertical blank period
ipu_active_bufid[2:0]	Input	IPU	Tell GPU3D which frame is free to be overwritten, one-hot coded

The vertical sync (VSYNC) from the display controller occurs after each frame is displayed, during the vertical blanking period.

One-hot encoded USE_BUFID signals are updated by GPU3D whenever there is a new rendered buffer available (and the previous value of USE_BUFID has already been sampled by the IPU/display controller), so it can change anywhere in the frame. It is resynchronized and sampled at the start of the vertical blanking period (rising edge of VSYNC) in display controller, while the currently finished buffer (ACTIVE_BUFID signals) is updated by display controller at the rising edge of VSYNC, ACTIVE_BUFID are then resynchronized and sampled by GPU3D after synchronizing to VSYNC rising edge.

38.4.5 Debug Bus and GPIO

This section describes the debug bug and GPIO.

38.4.5.1 Debug Bus

The GPU3D has a 32b debug bus output that is controlled by a 18b GPIO register. The 32b debug bus should be mapped to system GPIO. In i.MX51, this 32b bus is MUXed to 16b to reduce the number of external pins required. The MUXing of these pins is controlled by 2 bits in the GPIO register.

38.4.5.2 GPIO Register

The GPU3D supports 16 general-purpose input pins that can be configured as soft reset, debug mux select bits etc. The bus is implemented as a system register (whose offset is 0x2_FFFC from GPU3D base address) along with 2_bit debug bus control information. The registered inputs are semistatic and have no timing constraints associated with them.

Table 38-2 shows the GPIO register implementation.

Table 38-2. GPIO Register-—R/W 18 Bit (0x2_FFFC)

Field Name	Bits	Default	Description
Debug Bus Control	17–16	0x0	00 Output lower 16 bits 01 Output upper 16 bits 10 Toggle between outputting lower and upper 16 bits per 83 MHz clock 11 Unused
GPU3D GPIO Bus	15–0	0x0000	Debug Bus Control

Table 38-3 lists the GPU3D GPIO bus functions:

Table 38-3. GPU3D GPIO Bus Function

GPIO BIT	Debug Function	Reset/Misc Function
15	1’b0	Ignore SQ RTR for AHB
14	1’b0	Ignore VGT RTR for AHB
13	1’b0	Ignore CP RTR for AHB
12	1’b0	Ignore RTR for AHB
11	Sub-Block Select[3]	Ignore RTR
10	Sub-Block Select[2]	VGT Soft Reset
9	Sub-Block Select[1]	SC Soft Reset
8	Sub-Block Select[0]	CIB Soft Reset
7	1’b0	1’b0
6	1’b0	1’b1
5	Sub-Block Addr[5]	SX Soft Reset

Table 38-3. GPU3D GPIO Bus Function (continued)

GPIO BIT	Debug Function	Reset/Misc Function
4	Sub-Block Addr[4]	SQ Soft Reset
3	Sub-Block Addr[3]	RB Soft Reset
2	Sub-Block Addr[2]	MH Soft Reset
1	Sub-Block Addr[1]	PA Soft Reset
0	Sub-Block Addr[0]	CP Soft Reset

The functionality in the Reset/Misc Function is the same as the functionality in the RBBM_SOFT_RESET register and the RBBM_DEBUG register. Please see the AMD's "ATI Yamato Register Spec" for more details on this functionality. The bits are logically OR'd together.

There is latency built into the debug bus due to the registered daisy chain nature of the debug bus. From the time GPU3D GPIO is registered (count this as clock 1), it can take up to 10 clock cycles before valid debug bus data is resident in the debug bus register. This latency applies any time the state of GPU3D GPIO is changed.

38.5 Clocking Architecture

This section describes clock input and gating.

38.5.1 Clock input

The GPU3D design has twp clocks at top level: ***aclk_gpu*** for the graphics core and ***aclk_garb*** for the graphic memory (GMEM) and its arbiter (GARB). These two clocks are synchronous, meaning that they are in a single domain. No multicycle paths, latches, or negative edge flops are used in the design. (The CKGATER uses a negative latch, but that is a special cell.)

Both ***aclk_gpu*** and ***aclk_garb*** are synchronous and in phase with the core clock. They come from one source in CCM, but have a separate gating cell so that ***aclk_gpu,*** the clock to the graphics core can be removed independently of ***aclk_garb***, which keeps graphic memory accessible when the graphics core is shut down.

Both ***aclk_gpu*** and ***aclk_garb*** should be implemented with programmable dividers to run at full speed, synchronous-bus speed, and half-bus speed to meet system power use cases.

38.5.2 Clock Gating

GPU3D graphics core (GC) supports the following three levels of clock gating:

- Explicit removal of clocks to the GC core by the system under software control
- Designer controlled removal of clocks to logic blocks (controlled by internal CIB module)
- Instantiated clock gating by synthesis tool.

Explicit clock gating is invoked under system control and as such is transparent to the GPU3D. All clocks (***aclk_gpu*** and ***aclk_garb***) are removed. Prior to clock removal, the system should ensure that the GPU3D is idle (no outstanding bus traffic, no outstanding command activity in command buffer). Removing the clocks to the GPU3D does not impact internal state (unless power is also removed).

Designers can chose to remove clocks to blocks or sections of blocks based on those functions becoming idle. Power Management Override bits are supported which selectively disable designer instantiated clock gating. The GC powers up with the clock gating disabled with the exception of the AHB busclock and the GMEM clocks which power up with clock gating enabled. It is the responsibility of the driver to enable the clock gating functionality by disabling (clearing) the power management override bits (by setting 2 registers RBBM_PM_OVERRIDE1 and RBBM_PM_OVERRIDE2). The GARB instantiate default enabled clock gaters for GMEM clock for power saving, which can be overridden (dibbled) by pm_override[3:0] from the GC to the GARB (by register RBBM_PM_OVERRIDE2).

In particular, the output ‘idle’ signal (gpu_idle, active high) signal from the CIB should be used to clock gate the core clock signals (***aclk_gpu***) to reduce power consumption.

38.6 Reset

GPU3D has only an asynchronous reset port named hresetn at top level. This port is active low and resets the graphics core, graphic memory, and its arbiter.

CAUTION

The internal graphics core uses synchronous reset, which is propagated through the design as a standard synchronous signal. Therefore the clocks (**ack_gc** and **aclk_garb**) must be turned on when hard reset (hresetn) is asserted.

The supplied reset (hresetn) is passed through a reset conditioning circuit in the CIB. This conditioning circuit extends reset assertion long enough to guarantee that all internal circuits are reset correctly. This circuit counts 64 bus cycles after the de-assertion of reset so GPU3D need 64 bus cycles to complete a reset.

In addition to the top level hard reset (hresetn), there are soft resets for each internal block. These resets can be controlled by software (by register RBBM_SOFT_RESET).

38.7 Interrupts

The graphics core provides a single interrupt pin to the system (gpu_int_b). Once asserted the interrupt pin remains asserted (active low) until the CPU clears the IRQ bit in the RBBM_IRQ register.

In general interrupts are used to signal error conditions or to support frame buffer swap.

38.8 Memory Map

The GPU3D memory map is shown in Table 38-4. There are two memory ranges:

- AHB slave interface (graphics core space)
- AXI slave memory interface (graphics memory space).

Table 38-4. . GPU3D Memory Map

Description	Memory Address Base	Memory Address End
GC space (register, instruction)	0x3000_0000	0x3002_FFFF
GMEM space (128 Kbytes)	0x2000_0000	0x2002_0000

Chapter 39
High Speed Inter IC (HS-I^{2}C)

39.1 Introduction

The High Speed Inter IC (HS-I^{2}C) is bidirectional, two wire, serial bus interface module. The HS_I^{2}C is designed to be compatible with the standard Philips I^{2}C bus protocol, version 2.1. Refer to Figure 39-1 for the HS_I^{2}C module block diagram.

NOTE

Due to limited functionality, Freescale does not recommend users incorporate the HS-I2C module in future designs. For details on the limitations, refer to the i.MX51 IC Errata. Users should consider using the two standard I2C modules, which have no errata.

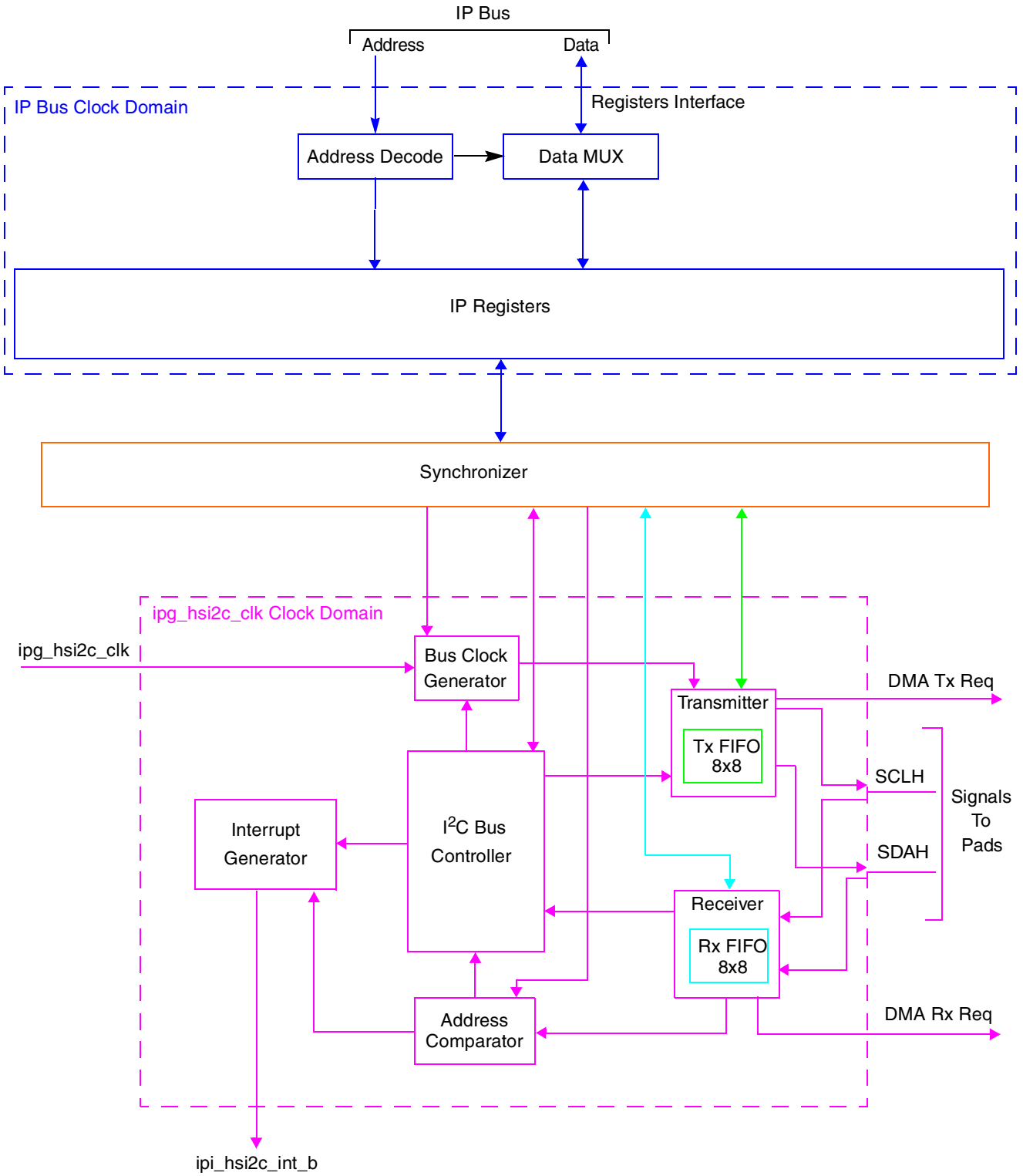


Figure 39-1. HS_I^{2}C Block Diagram

39.1.1 Overview

The HS_I^{2}C module provides simple and efficient way of 8-bit data transfers between I^{2}C bus compatible devices. The two bus lines: a serial data line (SDAH) and serial clock line (SCLH) provide data transfer rates upto 100 Kbits/s in Standard Mode, data rates upto 400 Kbits/s in Fast Mode and data rates upto 3.4 Mbits/s in High-Speed Mode. However, the data transfer rate depends upon the pad loading. The HS_I^{2}C module allows additional devices to be connected to the bus for system expansion and development. Refer to Figure 39-2 for connection of devices to the I^{2}C bus.

The HS_I^{2}C module provides a true multiple-master bus interface including arbitration and collision detection that prevents data corruption if multiple devices attempt to control the bus simultaneously.

The HS_I^{2}C incorporates 8x8 FIFO for transmit and 8x8 FIFO for receive purpose. The module IP interface supports the DMA data transfer operation to or from the HS_I^{2}C.

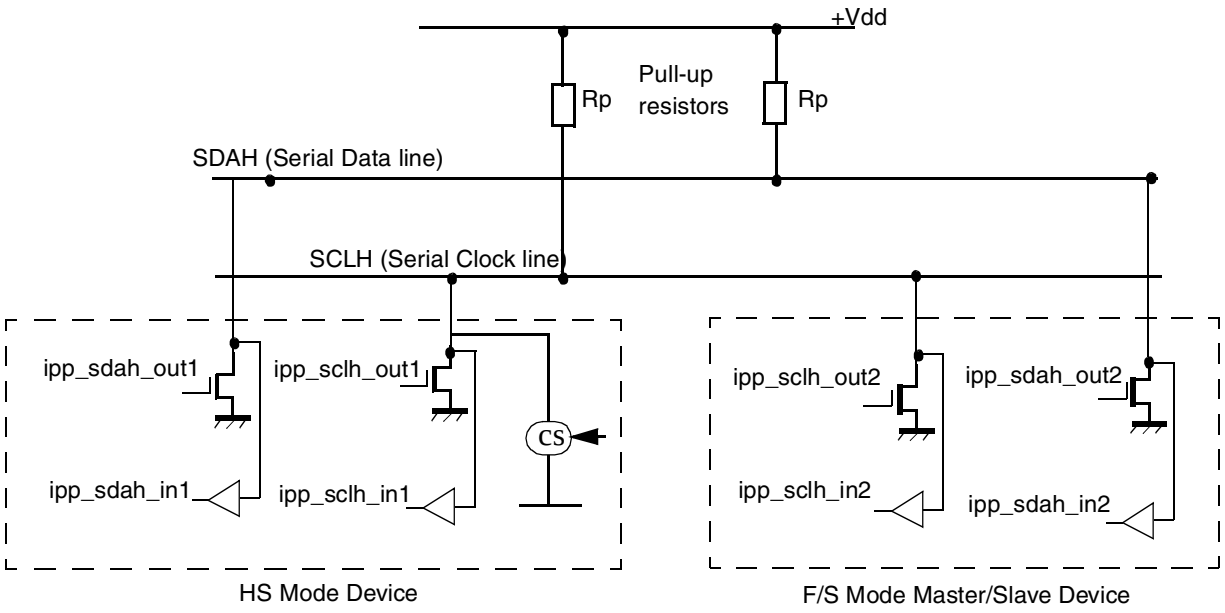


Figure 39-2. Connection of Devices to I^{2}C Bus

39.1.2 Features

The HS_I^{2}C module has the following key features.

- Compatible with the Philips I^{2}C bus standard version 2.1.
  - — Supports 7 and 10 bit addressing.
  - — Supports Standard Mode, Fast Mode and High-Speed Modes.
- Supports multiple master operation.
- Software programmable serial clock frequencies for Standard/Fast Mode and HS-Mode data transfer.

- Software option to select between High-Speed mode and Standard/Fast mode.
- Interrupt-driven, byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Slave address match interrupt
- Bus-busy detection
- Acknowledge bit generation/detection
- Supports DMA interface.
- Two 8x8 FIFO, one for Tx and one for Rx.

39.1.3 Modes of Operation

The HS_I^{2}C module primarily operates in 3 different functional modes.

39.1.3.1 Standard Mode

In this mode, HS_I^{2}C module supports the data transfer rates upto 100 Kbits/s.

39.1.3.2 Fast Mode

In Fast Mode data transfer rates of the order 400Kbits/s can be achieved. As per module operation, there is no special configuration required for Fast and Standard mode. It is the data transfer rate which distinguishes Standard and Fast mode.

39.1.3.3 High-Speed Mode

The data transfer rate as high as 3.4 Mbits/s can be achieved in this mode.

39.2 External Signal Description

There are two pins for HS_I^{2}C operation.

HS_I^{2}C module pins:

- SCLH—Bidirectional Clock Pin.
- SDAH—Bidirectional Data Pin.

For standard I^{2}C bus compliance, all devices connected to the SCLH and SDAH signals must have open-drain outputs. The logic AND function is exercised on both lines with external pull-up resistors in Standard/Fast mode operation. The resistor and CS pull up is used for SCLH line for HS-mode operation.

The module port signals that are connected to the pads are tabulated in Table 39-1, with relevant details.

Table 39-1. Pad Signal Properties

Name	Function	I/O	Reset	Pull Up
ipp_sclh_in	Serial Clock input	I	1	—
ipp_sclh_out	Serial Clock output	O	1	CS/Resister -Active
ipp_sclh_out_en	Serial Clock output enable	O	0	—
ipp_sdah_in	Serial Data input	I	1	—
ipp_sdah_out	Serial Data output	O	1	Resistor-Active
ipp_sdah_out_en	Serial Data output enable	O	0	—
ipp_hs_cs_cntrl	HS Mode current source pull up control	O	0	—
ipp_hs_cntrl	1) To adapt SDAH and SCLH input filters according to spike suppression requirement in HS 2) Slope control of SDLH and SCLH output stages in HS Mode	O	0	—

39.3 Memory Map and Register Definition

39.3.1 Memory Map

Table 39-2 is the memory map for the module.

39.3.2 Register Summary

Table 39-2. Block Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x0000(HISADR)	HS_I^{2}C Slave Address Register	R/W	0X0000	39.3.3.1/39-8
0xBASE+0x0004(HIMADR)	HS_I^{2}C Master Address Register	R/W	0x0000	39.3.3.2/39-9
0xBASE+0x0008(HICR)	HS_I^{2}C Control Register	R/W	0x0000	39.3.3.3/39-9
0xBASE+0x000C(HISR)	HS_I^{2}C Status Register	R/W	0x0002	39.3.3.4/39-12
0xBASE+0x0010(HIIMR)	HS_I^{2}C Interrupt Mask Register	R/W	0x00FF	39.3.3.5/39-13
0xBASE+0x0014(HITDR)	HS_I^{2}C Tx Data Register	W	0x0000	39.3.3.6/39-15
0xBASE+0x0018(HIRDR)	HS_I^{2}C Rx Data Register	R	0x0000	39.3.3.7/39-15
0xBASE+0x001C(HIFSFDR)	HS_I^{2}C F/S-Mode Frequency Divider Register	R/W	0x0000	39.3.3.8/39-16
0xBASE+0x0020(HIHSFDR)	HS_I^{2}C HS-Mode Frequency Divider Register	R/W	0X0000	39.3.3.9/39-17
0xBASE+0x0024(HITFR)	HS_I^{2}C Tx FIFO Control/Status Register	R/W	0x0000	39.3.3.10/39-18
0xBASE+0x0028(HIRFR)	HS_I^{2}C Rx FIFO Control/Status Register	R/W	0x0000	39.3.3.11/39-20
0xBASE+0x002C(HITDCR)	HS_I^{2}C Transmit Data Count register	R/W	0x00FF	39.3.3.12/39-21
0xBASE+0x0030(HIRDCR)	HS_I^{2}C Receive Data Count register	R/W	0x00FF	39.3.3.13/39-22

The conventions in Figure 39-3 and Table 39-3 serve as a key for the register summary and individual register diagrams.

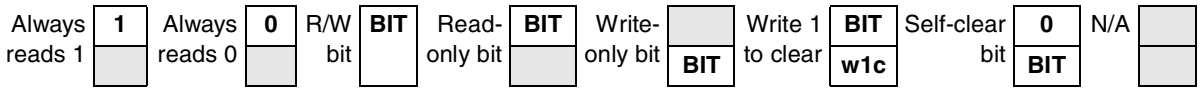


Figure 39-3. Key to Register Fields

Table 39-3 provides a key for register figures and tables and the register summary.

Table 39-3. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	

Table 39-3. Register Conventions (continued)

Convention	Description
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 39-4 shows the HS_I^{2}C register summary table.

Table 39-4. HS_I^{2}C register summary table

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x00(HISADR)	R	0	0	0	0	0	MSB_ADR			LSB_ADR							0
	W																
0xBASE+0x04(HIMADR)	R	0	0	0	0	0	MSB_ADR			LSB_ADR							0
	W																
0xBASE+0x08(HICR)	R	AUTO_RSTA	SAMC		HSM_EN	HS_MST_CODE			ADDR_MODE	HIIEN	MSTA	MTX	TXAK	RSTA	DMA_EN_TX	DMA_EN_RX	HIEN
	W																
0xBASE+0x0C(HISR)	R	0	0	0	0	SHS_MODE	SADDR_MODE	SRW	HIBB	RXAK	TDC_ZERO	RDC	BTD	HIAL	HIAAS	TDE	RDF
	W									w1c	w1c	w1c	w1c	w1c	w1c		
0xBASE+0x10(HIIMR)	R	0	0	0	0	0	0	0	0	MASK_RXAK	MASK_TDC	MASK_RDC	MASK_BTD	MASK_AL	MASK_AAS	MASK_TDE	MASK_RDF
	W																
0xBASE+0x14(HITDR)	R	0	0	0	0	0	0	0	0								
	W									TX_DATA							

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x18(HIRDR)	R	0	0	0	0	0	0	0	0	RX_DATA							
	W																
0xBASE+0x1C (HIFSFDR)	R	0	0	0	0	0	0	0	0	0	0	FSICR					
	W																
0xBASE+0x20(HIHSFDR)	R	0	0	0	0	0	0	0	0	0	0	HSICR					
	W																
0xBASE+0x24(HITFR)	R	0	0	0	0	TFC				0	0	0	TFWM			TFLSH	TFEN
	W																
0xBASE+0x28 (HIRFR)	R	0	0	0	0	RFC				0	0	0	RFWM			RFLSH	RFEN
	W																
0xBASE+0x2C (HITDCR)	R	0	0	0	0	0	0	TDC_RSTA	TDC_EN	TDC							
	W																
0xBASE+0x30 (HIRDCR)	R	0	0	0	0	0	0	RDC_RSTA	RDC_EN	RDC							
	W																

39.3.3 Register Descriptions

This section contains the detailed register descriptions for the HS_I^{2}C registers in address order.

39.3.3.1 HS_I^{2}C Slave Address Register (HISADR)

Figure 39-4 shows the HS_I^{2}C Slave Address Register; Table 39-5 provide its field descriptions.

0xBASE+0x00(HISADR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	MSB_ADR			LSB_ADR							0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-4. HS_I^{2}C Slave Address Register

The HISADR contains the 7 and 10 bit slave addresses of the HS_I^{2}C module.The module responds to this address when addressed as a slave in either 7 or 10 bit addressing mode.The HISADR can be programmed only when HICR[HIEN] bit is set to 0.The register is not reset by a software reset.

NOTE

The slave address is not the address sent on the bus during the address transfer.

Table 39-5. HS_I²C Slave Address Register Field Descriptions

Field	Description
15–11	Reserved
10-8 MSB_ADR	MSB of 10 bit slave address: Contains the three MSB bits of the 10 bit slave address
7–1 LSB_ADR	LSB of the 10 bit slave address: Contains the LSB of the slave address in 10 bit addressing mode. This bit also serves as the 7 bit slave address when the module is selected for slave mode in 7 bit addressing mode. Slave mode is the default HS_I²C mode for an address match on the bus.
0	Reserved

39.3.3.2 HS_I²C Master Address Register (HIMADR)

Figure 39-5 shows the HS_I²C Master Address Register; Table 39-6 provide its field descriptions.

0xBASE+0x04(HIMADR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	MSB_ADR			LSB_ADR							0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-5. HS_I²C Master Address Register

The HIMADR contains the address of the slave to be addressed when HS_I²C is configured as master in 7 and 10 bit addressing modes. The module sends out this address on the I²C bus during address transfer for 7 and 10 bit addressing modes. The register is not reset by a software reset.

Table 39-6. HS_I²C Master Address Register Field Description

Field	Description
15–11	Reserved
10-8 MSB_ADR	MSB of 10 bit master address: Contains the three MSB bits of the 10 bit master address.
7–1 LSB_ADR	LSB of the 10 bit master address: Contains the LSB of the master address in 10 bit addressing mode. This bit also serves as the 7 bit master address when the module is bus master in 7 bit addressing mode.
0	Reserved

39.3.3.3 HS_I²C Control Register (HICR)

The HICR is used to enable the HS_I²C module and the HS_I²C interrupt. It also contains bits that govern module operation as a slave or a master. Figure 39-6 shows the HS_I²C Control Register; Table 39-7 provides its field descriptions.

0xBASE+0x08(HICR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	AUTO_RSTA	SAMC		HSM_EN	HS_MST_CODE			ADDR_MODE	HIIEN	MSTA	MTX	TXAK	RSTA	DMA_EN_TX	DMA_EN_RX	HIEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-6. HS_I^{2}C Control Register

Table 39-7. HS_I^{2}C Control Register Field Description

Field	Description
15 AUTO_RSTA	Auto Repeated Start: This bit will control whether auto repeated start is generated or not when HS_I^{2}C is configured for master transmit operation and DMA and FIFO or only FIFO is enabled and ACK is not received for particular address or data phase. If this bit is set auto repeated start is generated when no ACK is received. 0 Auto repeated start is not generated. (Default) 1 Auto repeated start is generated.
14-13 SAMC	Slave Address Mode Control: These bits control whether the HS_I^{2}C will respond to 7 or 10 bit addressing or both when configured as slave. These bits can be programmed only when HICR[HIEN] is 0. 00 - HS_I^{2}C can be selected in both 7 and 10 bit addressing mode.(Default) 01 - HS_I^{2}C can be selected in 7 bit addressing mode only 10 - HS_I^{2}C can be selected in 10 bit addressing mode only 11 - Reserved
12 HSM_EN	High Speed Mode Enable: Selects the High Speed mode or Fast/Standard mode operation for the module in master mode. The HSM_EN bit should be set to select the High Speed mode operation. After the transmission of the master code in F/S mode and the HS_I^{2}C does not loose the arbitration, automatic switching from F/S to HS mode takes place. 0 Fast/Standard mode operation. (Default) 1 High Speed mode operation.
11-9 HS_MST_CODE	High Speed Master Code: Controls the master code for high-speed mode data transfer. These bits get appended to MSB five bits of the HS master code 00001 and final 8 bit code is sent as the high speed master code. Possible value range for these bits is 001-111. These bits can be programmed only when HICR[MSTA] is 0. **Note:** Combination 00001000 is reserved for the test and debug purposes.
8 ADDR_MODE	Address Mode select: This bit controls the address mode selected for the data transfer in master mode. 0 7 bit addressing is selected (Default) 1 10 bit addressing mode is selected.
7 HIIEN	HS_I^{2}C interrupt enable. 0 HS_I^{2}C module interrupt is disabled.(Default) 1 HS_I^{2}C module interrupt is enabled. An interrupt is generated whenever any one of the following status flags are set and corresponding mask bits in Interrupt Mask Register (i.e. HIIMR) are cleared. • One byte transfer is completed either in receive or transmit mode. (the interrupt is set at the falling edge of the ninth clock). • An address is received that matches its own specific address in slave-receive mode. • Arbitration is lost. • Tx FIFO transmit data empty • RX FIFO receive data full

Table 39-7. HS_I²C Control Register Field Description (continued)

Field	Description
6 MSTA	Master/slave mode select bit. This bit controls whether the HS_I²C is in master or slave mode. For master mode operation this bit should be set to 1. Changing MSTA from 1 to 0 generates a STOP condition on the bus and selects slave mode. If MSTA is changed from 0 to 1, the module generates START condition on the bus and selects master mode. When HS_I²C is configured as master and looses bus arbitration, MSTA is cleared automatically without generating a STOP signal on the bus. If MSTA bit is cleared while data transfer is going on, stop will be generated at byte boundary only. **Note:** Module ipg_hsi2c_clk clock should be ON for writing to the MSTA bit. 0 Slave mode.(Default) 1 Master mode.
5 MTX	Transmit/receive mode select bit. Selects the direction of master and slave transfers. 0 Receive.(Default) When a slave is addressed, the software should set MTX according to the slave read/write bit in the HS_I²C status register (HISR[SRW]). 1 Transmit. In master mode, MTX should be set according to the type of transfer required.
4 TXAK	Transmit acknowledge enable. Specifies the value driven onto SDAH during acknowledge cycles for both master and slave receivers. **Note:** This bit is effective only when HS_I²C module act as a receiver. 0 An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte of data.(Default) 1 No acknowledge signal response is sent (that is, the acknowledge bit = 1).
3 RSTA	Repeated start. Initiates the repeated start on the I²C bus in master mode. The RSTA is self clearing bit which gets cleared at the data or address byte boundary. The core needs to read this bit before programming it again. **Note:** Attempting a repeated start without bus mastership causes loss of arbitration 0 No repeated start.(Default) 1 Generates a repeated START condition
2 DMA_EN_TX	DMA Enable for TX. DMA_EN_TX specifies whether the data transfer is through the core or the DMA. If enabled, DMA Tx request is generated which initiates the DMA write cycle for HITDR register. When disabled, only the core can access HITDR data register and DMA Tx request is not generated. Hardware or software reset clears this bit. 0 DMA data interface disabled (Default) 1 DMA data interface enabled
1 DMA_EN_RX	DMA Enable for RX. DMA_EN_RX specifies whether the data transfer is through the core or DMA. If enabled, DMA Rx request is generated which initiates DMA read cycle for HIRDR register. When disabled, only the core can access data register or RX FIFO and DMA Rx request is not generated. Hardware or software reset clears this bit. 0 DMA data interface disabled (Default) 1 DMA data interface enabled
0 HIEN	HS_I²C enable. This bit controls the software reset of the entire HS_I²C module. Resetting the bit generates an internal reset to the module. If the module is enabled in between a byte transfer on the I²C bus, slave mode ignores the current bus transfer and starts operating when the next start condition is detected. Master mode is not aware that the bus is busy; so initiating a start cycle may corrupt the current bus cycle, ultimately causing either the current master or the HS_I²C module to lose arbitration. After which, bus operation returns to normal. This bit must be set before proceeding to programming of the other bits/registers. 0 The module is disabled (Default) 1 The HS_I²C module is enabled

39.3.3.4 HS_I²C Status Register (HISR)

The HISR contains bits that indicate transaction direction and status. Figure 39-7 shows the HS_I²C Status Register; and Table 39-8 provides its field descriptions.

Access: User read/write

0xBASE+0x0C(HISR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	SHS_MODE	SADDR_MODE	SRW	HIBB	RXAK	TDC_ZERO	RDC_ZERO	BTD	HIAL	HIAAS	TDE	RDF
W									w1c	w1c	w1c	w1c	w1c	w1c		
RESET	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 39-7. HS_I²C Status Register

Table 39-8. HS_I²C Status Register Field Description

Field	Description
15–12	Reserved
11 SHS_MODE	Slave high speed mode. When HS_I²C is selected as slave, this bit indicates whether the module is selected for High-Speed mode or Fast/Standard mode data transfer. 0 HS_I²C is selected for Fast/Standard data transfer (Default). 1 HS_I²C is selected for High-Speed mode data transfer.
10 SADDR_MODE	Slave address mode. When HS_I²C is selected as slave, this bit indicates whether 7 or 10 bit addressing is used. 0 7 bit addressing is used (Default). 1 10 bit addressing is used.
9 SRW	Slave read/write. When the HS_I²C is addressed as a slave, HIAAS is set, and the slave read/write bit (SRW) indicates the value of the R/W command bit of the calling address sent from the master. SRW is valid only when a complete transfer has occurred, no other transfers have been initiated, and the HS_I²C module is a slave and has an address match. 0 Slave receive, master writing to slave (Default) 1 Slave transmit, master reading from slave
8 HIBB	HS_I²C bus busy. Indicates the status of the bus. 0 Bus is idle. When STOP condition is detected, HIBB is cleared (Default) 1 Bus is busy. When START condition is detected, HIBB is set.
7 RXAK	Received acknowledge. This bit is set when a NOACK value is received from the SDAH input for the acknowledge for the acknowledge cycle on I²C bus. This bit should be cleared by writing 1 to it.When HS_I²C is generating ACK or NOACK, this bit is invalid. 0 No “NOT ACK” bit received (Default) 1 A “No acknowledge” signal was detected at the ninth clock
6 TDC_ZERO	Transmit Data Counter Zero. This bit is set when HITDCR[TDC_EN] =1 and the TX data count HITDCR[TDC_ZERO] reaches 0. The TDC_ZERO bit must be cleared by software by writing a “1” to it 0 Transmit data counter (HITDCR[TDC]) is not zero (Default) 1 Transmit data counter (HITDCR[TDC]) is zero

Table 39-8. HS_I^{2}C Status Register Field Description (continued)

Field	Description
5 RDC_ZERO	Receive Data Counter Zero. This bit is set when HIRDCR[RDC_EN] =1 and the RX data count HIRDCR[RDC] reaches 0.The RDC_ZERO bit must be cleared by software by writing a “1” to it 0 Receive data counter (HIRDCR[RDC]) is not zero (Default) 1 Receive data counter (HIRDCR[RDC]) is zero
4 BTD	Byte transfer done. This bit is set when a byte of address/data has been transferred on the I^{2}C bus. The BTD bit must be cleared by software by writing a “1” to it. 0 Byte Transfer not Complete (Default) 1 Byte Transfer is complete, and set by the falling edge of the ninth clock of a byte transfer
3 HIAL	HS_I^{2}C Arbitration lost. This bit is set by HS_I^{2}C in the following circumstances (This bit must be cleared by writing a “1” to it): • SDAH input sampled low when the master drives high during an address or data-transmit cycle. • SDAH input sampled low when the master drives high during the acknowledge bit of a data-receive cycle. For the above two cases, the bit is set at the falling edge of 9th SCLH clock during the ACK cycle. • A start cycle is attempted when the bus is busy. • A repeated start cycle is requested in slave mode. • A stop condition is detected when the master did not request it. **Note:** Software cannot set the bit. 0 No arbitration lost (Default) 1 Arbitration is lost.
2 HIAAS	HS_I^{2}C addressed as a slave bit. The CPU is interrupted if the interrupt enable (HICR[HIIEN]) is set. The core must check the slave read/write bit (SRW) and set its TX/RX mode accordingly. This Bit must be cleared by writing a 1 to it. 0 Not addressed (Default) 1 Addressed as a slave. Set when its own address (HISADR) matches the calling address on I^{2}C bus.
1 TDE	Transmit Data Empty: TDE flag is set when the data level in TX FIFO is less than or equal to the transmit FIFO empty watermark value. When set, TDE indicates that data should be written to the TX FIFO. If HICR[HIIEN] bit is set and HIIMR[TDE] bit is zero, transmit data interrupt will be generated. TDE flag is cleared automatically when the data level in the TX FIFO becomes more than the transmit FIFO empty watermark level. Hardware reset, software reset, and Transmit FIFO flush operation set TDE. 0 Data level in TX FIFO is above transmit FIFO empty watermark level 1 Data level in TX FIFO is less than or equal to the transmit FIFO empty watermark value (Default)
0 RDF	Receive Data Full: RDF flag is set whenever the data level in RX FIFO equals or exceeds the selected watermark threshold. If HICR[HIIEN] bit is set and HIIMR[RDF] bit is zero, receive data interrupt will be generated. This bit is cleared automatically whenever the data level in RX FIFO is below the watermark threshold. Hardware reset, software reset, and Receive FIFO flush operation clears this flag. 0 data level in RX FIFO is below the selected watermark threshold (Default) 1 data level in RX FIFO has reached the selected watermark threshold

39.3.3.5 HS_I^{2}C Interrupt Mask Register (HIIMR)

The HIIMR contains the interrupt mask bits for various HS_I^{2}C interrupt conditions. Figure 39-8 shows the register: Table 39-9 describes the HIIMR fields.

Access: User Read/Write

0xBASE+0x10 (HIIMR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	MASK_RXAK	MASK_TDC	MASK_RDC	MASK_BTD	MASK_AL	MASK_AAS	MASK_TDE	MASK_RDF
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 39-8. HS_I^{2}C Interrupt Mask Register (HIIMR)

Table 39-9. HS_I^{2}C Interrupt Mask Register Field Descriptions

Field	Description
15–8	Reserved
7 MASK_RXAK	Mask Received Acknowledge Interrupt: This bit masks the received acknowledge interrupt. If this bit is cleared and HICR[HIIEN] is set and whenever acknowledge is not received, interrupt is generated. 0 Received Acknowledge interrupt generated if HICR[HIIEN] is set 1 Mask received acknowledge interrupt (Default)
6 MASK_TDC	Mask Transmit Data Count Zero Interrupt: This bit masks transmit data count zero interrupt. If this bit is cleared and HICR[HIIEN] is set and whenever transmit data count is zero, interrupt is generated. 0 Transmit data count interrupt generated if HICR[HIIEN] is set 1 Mask transmit data count interrupt (Default)
5 MASK_RDC	Mask Receive Data Count Zero Interrupt: This bit masks receive data count zero interrupt. If this bit is cleared and HICR[HIIEN] is set and whenever receive data count is zero, interrupt is generated. 0 Receive data count interrupt generated if HICR[HIIEN] is set 1 Mask receive data count interrupt (Default)
4 MASK_BTD	Mask Data Transfer Interrupt. This bit masks byte transfer done interrupt. If this bit is cleared and HICR[HIIEN] is set and whenever one byte transfer is completed either in receive or transmit mode, interrupt is generated at the falling edge of the ninth clock. 0 Data transfer interrupt generated if HICR[HIIEN] is set. 1 Mask data transfer interrupt (Default)
3 MASK_AL	Mask Arbitration Lost Interrupt. This bit masks the arbitration lost interrupt. If this bit is cleared and HS_I^{2}C module looses bus arbitration (i.e HISR[HIAL] is set) and HICR[HIIEN] is set, interrupt will be generated. 0 Arbitration lost interrupt is generated if HICR[HIIEN] is set. 1 Mask arbitration lost interrupt (Default)
2 MASK_AAS	Mask Addressed As Slave Interrupt. This bit masks the HS_I^{2}C addressed as slave interrupt. If a address is received that matches its own specific address in slave-receive mode (i.e. HISR[HIAAS] is set) and HICR[HIIEN] is set and HIIMR[MASK_AAS] is cleared, interrupt will be generated. 0 HS_I^{2}C addressed as slave interrupt is generated if HICR[HIIEN] is set 1 Mask addressed as slave interrupt (Default).

Table 39-9. HS_I²C Interrupt Mask Register Field Descriptions (continued)

Field	Description
1 MASK_TDE	Mask Transmit Data Empty Interrupt. This bit masks the Tx FIFO transmit data empty interrupt. If this bit is cleared and HISR[TDE] is set and HICR[HIIEN] bit is set, interrupt will be generated. 0 Transmit data empty interrupt is generated if HICR[HIIEN] is set 1 Mask transmit data empty interrupt (Default)
0 MASK_RDF	Mask Receive Data Full Interrupt. This bit masks the Rx FIFO receive data full interrupt. If this bit cleared and HISR[RDF] is set and HICR[HIIEN] bit is set, interrupt will be generated. 0 Receive data full interrupt generated if HICR[HIIEN] is set 1 Mast receive data full interrupt (Default)

39.3.3.6 HS_I²C Tx Data Register (HITDR)

The HITDR register serves as the transmit data register in transmit mode. This register in write only register. Figure 39-9 shows the HS_I²C Tx Data Register; Table 39-10 provides its field descriptions.

0xBASE+0x14(HITDR) Access: User Write-only

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0								
W									TX_DATA							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-9. HS_I²C Tx Data Register

Table 39-10. HS_I²C Tx Data Register Field Descriptions

Field	Description
15–8	Reserved
7–0 TX_DATA	TX Data Byte. Holds the next data byte to be transferred. Software writes the next data byte to be transmitted if Tx FIFO is disabled.

39.3.3.7 HS_I²C Rx Data Register (HIRDR)

The HIRDR register serves as receive data register in receive mode. This register is read only register. Figure 39-10 shows the HS_I²C Rx Data Register; Table 39-11 provides its field descriptions.

Access: User Read-only

0xBASE+0x18(HIRDR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	RX_DATA							
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-10. HS_I²C Rx Data Register

Table 39-11. HS_I²C Rx Data Register Field Description

Field	Description
15–8	Reserved
7–0 RX_DATA	RX Data Byte. Holds the last data byte received. Software should read the data byte received if Rx FIFO is disabled.

39.3.3.8 HS_I²C F/S-Mode Frequency Divide Register (HIFSFDR)

The HIFSFDR provides a programmable prescaler to configure the clock for bit-rate selection for Fast/Standard mode HS_I²C operation. The register does not get reset by software reset. Figure 39-11 shows the HS_I²C F/S-Mode Frequency Register; Table 39-12 provides its field descriptions.

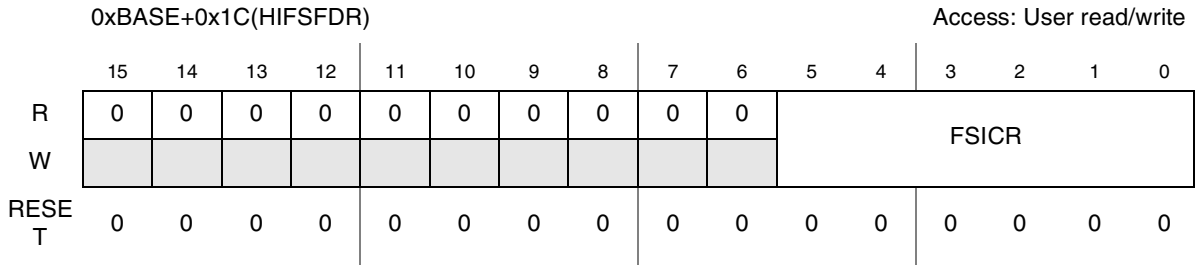


Figure 39-11. HS_I²C F/S Mode Frequency Divide Register

Table 39-12. HS_I²C F/S-Mode Frequency Divide Register Field Descriptions

Field	Description
15–6	Reserved
5–0 FSICR	F/S-Mode I²C clock rate. Prescales the clock for bit-rate selection. Due to potentially slow SCLH and SDAH rise and fall times, bus signals are sampled at the prescaler input frequency. The serial bit clock or prescaler output frequency is equal to module input ipg_hsi2c_clk clock divided by the divider shown in Table 39-13. **Note:** I²C protocol supports bit rates up to 400 Kbits/s in F/S-Mode. The FSICR bits need to be programmed in accordance with this constraint. The FSICR divider value will be used to generate the start and master code transmission in the beginning HS-Mode operation.

Table 39-13. HIFSFDR Register Field Values

FSIC R	Divider	FSIC R	Divider	FSIC R	Divider	FSIC R	Divider
0x00	26	0x10	256	0x20	34	0x30	384
0x01	28	0x11	288	0x21	36	0x31	416
0x02	30	0x12	320	0x22	38	0x32	448
0x03	32	0x13	352	0x23	40	0x33	480
0x04	40	0x14	512	0x24	56	0x34	768
0x05	44	0x15	576	0x25	60	0x35	832
0x06	48	0x16	640	0x26	64	0x36	896
0x07	52	0x17	704	0x27	68	0x37	960
0x08	72	0x18	1024	0x28	104	0x38	1536
0x09	80	0x19	1152	0x29	112	0x39	1664
0x0A	88	0x1A	1280	0x2A	120	0x3A	1792
0x0B	96	0x1B	1408	0x2B	128	0x3B	1920
0x0C	128	0x1C	2048	0x2C	192	0x3C	3072
0x0D	144	0x1D	2304	0x2D	208	0x3D	3328
0x0E	160	0x1E	2560	0x2E	224	0x3E	3584
0x0F	176	0x1F	2816	0x2F	240	0x3F	3840

39.3.3.9 HS_I²C HS-Mode Frequency Divide Register (HIHSFDR)

The HIHSFDR provides a programmable prescaler to configure the clock for bit-rate selection for HS-Mode HS_I²C operation. The register does not get reset by software reset. Figure 39-12 shows the HS_I²C HS-Mode Frequency Register; Table 39-14 provides its field descriptions.

Access: User read/write

0xBASE+0x20(HIHSFDR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	HSICR					
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-12. HS_I²C HS-Mode Frequency Divider Register

Table 39-14. HS_I²C HS-Mode Frequency Divide Register Field Description

Field	Description
15–6	Reserved
5–0 HSICR	HS-Mode I²C clock rate. Prescales the clock for bit-rate selection. Due to potentially slow SCLH and SDAH rise and fall times, bus signals are sampled at the prescaler input frequency. The serial bit clock or the prescaler output frequency is equal to module input ipg_hsi2c_clk clock divided by the divider shown in Table 39-15. **Note:** I²C protocol supports bit rates up to 3.4 Mbits/s in HS-Mode. The HSICR bits need to be programmed in accordance with this constraint.

Table 39-15. HIHSFDR Register Field Values

HSI CR	Divider	HSI CR	Divider	HSI CR	Divider	HSI CR	Divider
0x00	16	0x10	224	0x20	24	0x30	352
0x01	18	0x11	256	0x21	26	0x31	384
0x02	20	0x12	288	0x22	28	0x32	416
0x03	22	0x13	320	0x23	30	0x33	448
0x04	28	0x14	448	0x24	44	0x34	704
0x05	32	0x15	512	0x25	48	0x35	768
0x06	36	0x16	576	0x26	52	0x36	832
0x07	40	0x17	640	0x27	56	0x37	896
0x08	56	0x18	896	0x28	88	0x38	1408
0x09	64	0x19	1024	0x29	96	0x39	1536
0x0A	72	0x1A	1152	0x2A	104	0x3A	1664
0x0B	80	0x1B	1280	0x2B	112	0x3B	1792
0x0C	112	0x1C	1792	0x2C	176	0x3C	2816
0x0D	128	0x1D	2048	0x2D	192	0x3D	3072
0x0E	144	0x1E	2304	0x2E	208	0x3E	3328
0x0F	160	0x1F	2560	0x2F	224	0x3F	3584

39.3.3.10 HS_I²C Tx FIFO Register (HITFR)

The HITFR contains the Tx FIFO control and status bits. Figure 39-13 shows the register: Table 39-16 describes the HITFR fields.

0xBASE+0x24(HITFR) Access: User Read/Write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	TFC				0	0	0	TFWM			TFLSH	TFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-13. HS_I²C Tx FIFO Register (HITFR)

Table 39-16. HS_I²C Tx FIFO Register Field Descriptions

Field	Description
15–12	Reserved
11–8 TFC	Transmit FIFO Counter. These bits indicate the number of data words in the Transmit FIFO. Hardware reset, software reset or Transmit FIFO flush operation clears TFC bits. 0000 = 0 byte in transmit FIFO (Default) 0001 = 1 byte in transmit FIFO 0010 = 2 bytes in transmit FIFO .... 0111 = 7 bytes in transmit FIFO 1000 = 8 bytes in transmit FIFO 1001 to 1111 = Reserved
7 – 5	Reserved
4 - 2 TFWM	Transmit FIFO Empty WaterMark. These bits control the threshold at which the HISR[TDE] bit is set when TX FIFO is enabled. The TDE flag is set whenever the data level in the TX FIFO is less than or equal to the selected threshold. Hardware or software reset clears these bits. 000 = TDE is set, when the TX FIFO has 0 data word. (Default) 001 = TDE is set, when the TX FIFO has less than or equal to 1 data byte. 010 = TDE is set, when the TX FIFO has less than or equal to 2 data bytes. .... 111 = TDE is set, when the TX FIFO has less than or equal to 7 data bytes.
1 TFLSH	Transmit FIFO Flush. This bit controls the Transmit FIFO flush operation. Writing this bit as 1 would reset all the TX FIFO logic and hence reset the TX FIFO if enabled. Writing this bit as 0 has no effect. Once set, this bit would get cleared automatically when the TX FIFO logic is completely reset. After TX FIFO flush operation, the software should ensure that this bit is cleared before starting any operation involving TX FIFO. TFLSH bit should only be set when HS_I²C is not in transmit mode. Hardware or software reset clears this bit. 0 No action (Default) 1 TX FIFO logic is reset
0 TFEN	Transmit FIFO Enable. This bit enables the transmit FIFO. Hardware or software reset clears this bit. 0 Transmit FIFO disabled (Default) 1 Transmit FIFO enabled

39.3.3.11 HS_I²C Rx FIFO Register (HIRFR)

The HIRFR contains the Rx FIFO control and status bits. Figure 39-14 shows the register: Table 39-17 describes the HIRFR fields.

Access: User Read/Write

0xBASE+0x28 (HIRFR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	RFC				0	0	0	RFWM			RFLSH	RFEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 39-14. HS_I²C Rx FIFO Register (HIRFR)

Table 39-17. HS_I²C Rx FIFO Register Field Descriptions

Field	Description
15–12	Reserved
11–8 RFC	Receive FIFO Counter. These bits indicate the number of data words in the Receive FIFO. Hardware reset, software reset or Receive FIFO flush operation clears RFC bits. 0000 = 0 byte in receive FIFO (Default) 0001 = 1 byte in receive FIFO 0010 = 2 bytes in receive FIFO .... 0111 = 7 bytes in receive FIFO 1000 = 8 bytes in receive FIFO 1001 to 1111 = Reserved
7–5	Reserved
4–2 RFWM	Receive FIFO Full WaterMark. These bits control the threshold over which the HISR[RDF] bit is set when the RX FIFO is enabled. The RDF flag is set whenever the data level in the RX FIFO is greater than or equal to the selected threshold. Hardware or software reset clears this bit. 000 = RDF is set when the RX FIFO has 1 data byte (Default) 001 = RDF is set when the RX FIFO has 2 data bytes. 010 = RDF is set when the RX FIFO has 3 data bytes. .... 111 = RDF is set when the RX FIFO has 8 data bytes.

Table 39-17. HS_I²C Rx FIFO Register Field Descriptions (continued)

Field	Description
1 RFLSH	Receive FIFO Flush. This bit controls the Receive FIFO flush operation. Writing this bit as 1 would reset all the RX FIFO logic and hence reset the RX FIFO if enabled. Writing this bit as 0 has no effect. Once set, this bit would get cleared automatically when the RX FIFO logic is completely reset. After RX FIFO flush operation, the software should ensure that this bit is cleared before starting any operation involving RX FIFO. RFLSH bit should only be set when HS_I²C is not in receive mode. Hardware or software reset clears this bit. 0 No action (Default) 1 RX FIFO logic is reset
0 RFEN	Receive FIFO Enable This bit enables the Receive FIFO. Hardware or software reset clears this bit. 0 Receive FIFO disabled (Default) 1 Receive FIFO enabled

39.3.3.12 HS_I²C Tx Data Count Register (HITDCR)

The HITDCR contains the Tx data counter and the control bit. Figure 39-15 shows the register: Table 39-18 describes the HITDCR fields.

0xBASE+0x2C (HITDCR) Access: User Read/Write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	TDC_RSTA	TDC_EN	TDC							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 39-15. HS_I²C Tx Data Count Register (HITDCR)

Table 39-18. HS_I²C Tx Data Count Register Field Description

Field	Description
15–10	Reserved
9 TDC_RSTA	Transmit Data Count Repeated Start. This bit is used to program whether a STOP condition or Repeated Start Condition is to be initiated after transmitting TDC number of bytes. This bit will come into effect only when HS_I²C is in master mode and HITDCR[TDC_EN] bit is set. 0 STOP condition would be initiated on the I²C bus. The module would switch to slave mode clearing the HICR[MSTA] bit (Default) 1 Repeated Start condition would be initiated on I²C bus.
8 TDC_EN	Transmit Data Counter Enable: This bit enables the transmit data counter operation. The TDC_EN should only be set, when HS_I²C is configured for master Tx operation. 0 Tx data counter is disabled. (Default) 1 Tx data counter is enabled.
7–0 TDC	Transmit Data Counter: These bits contain the number of data to be transmitted. It is a down counter and upon reaching zero, HISR[TDC_ZERO] status flag will be set. The transmit data counter values should be set only once for required number of data transfers. 00000000 = All the bytes have been transmitted. 00000001 = 1 byte needs to be transmitted 00000010 = 2 bytes need to be transmitted. .... 11111110 = 254 bytes needs to be transmitted 11111111 = 255 bytes needs to be transmitted (Default)

39.3.3.13 HS_I²C Rx Data Count Register (HIRDCR)

The HIRDCR contains the Rx data counter and the control bit. Figure 39-16 shows the register: Table 39-19 describes the HIRDCR fields.

Access: User Read/Write

0xBASE+0x30 (HIRDCR)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	RDC_RSTA	RDC_EN	RDC							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 39-16. HS_I²C Rx Data Count Register (HIRDCR)

Table 39-19. HS_I²C Rx Data Count Register Field Description

Field	Description
15–10	Reserved
9 RDC_RSTA	Receive Data Count Repeated Start. This bit is used to program whether a STOP condition or Repeated Start Condition is to be initiated after receiving RDC number of bytes. This bit will come into effect only when HS_I²C is in master mode and HIRDCR[RDC_EN] bit is set. 0 STOP condition would be initiated on the I²C bus. The module would switch to slave mode clearing the HICR[MSTA] bit (Default). 1 Repeated Start condition would be initiated on I²C bus

Table 39-19. HS_I^{2}C Rx Data Count Register Field Description

Field	Description
8 RDC_EN	Receive Data Counter Enable: This bit enables the receive data counter operation. The RDC_EN should only be set, when HS_I^{2}C is configured for master Rx operation. 0 Rx data counter is disabled (Default) 1 Rx data counter is enabled
7–0 RDC	Receive Data Counter: These bits contain the number of data to be received. It is a down counter and upon reaching zero, HISR[RDC_ZERO] status flag will be set. Rx data counter values should only be set once for the required number of data receive. 00000000 = All the bytes have been received. 00000001 = 1 byte needs to be received 00000010 = 2 bytes need to be received .... 11111110 = 254 bytes needs to be received 11111111 = 255 bytes needs to be received (Default)

39.4 Functional Description

39.4.1 HS_I^{2}C System Configuration

After completion of reset, the HS_I^{2}C module defaults to slave receive mode. When HS_I^{2}C module is not operating as a master or responding to a slave transmit address, the module will default to the slave receiver state.

39.4.2 I^{2}C Protocol

The I^{2}C communication protocol consists of six components, as follows:

- START
- Data Source/Recipient
- Data Direction
- Slave Acknowledge
- Data Acknowledge
- STOP

Refer to Figure 39-17 for the I^{2}C standard communication protocol, as defined in the following sections.

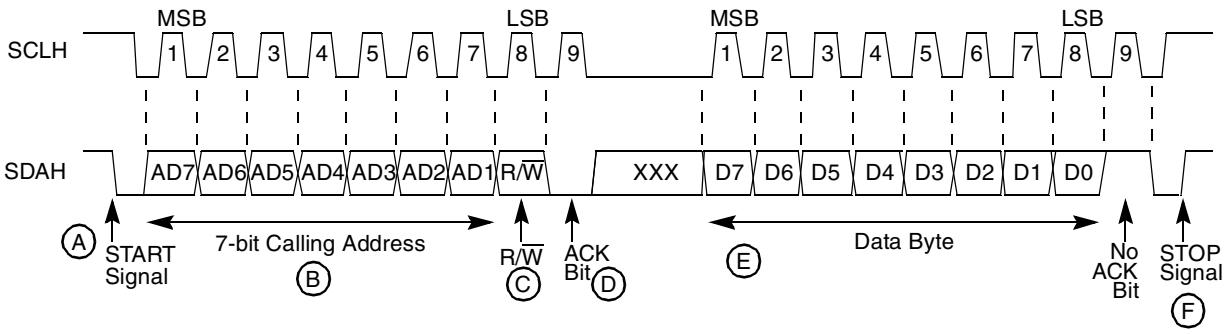


Figure 39-17. I²C Standard Communication Protocol

39.4.2.1 START Signal

When no other device is a bus master (both SCLH and SDAH lines are at logic high), a device can initiate communication by sending a START signal (Refer to A in Figure 39-17). A START signal is defined as a high-to-low transition of SDAH while SCLH is high. This signal denotes the beginning of a data transfer (each data transfer can be several bytes long) and awakens all slaves.

39.4.2.2 Slave Address Transmission

The master sends the slave address in the first byte after the START signal (B). After the seven-bit calling address, it sends the R/$\overline{W}$ bit (C), which decides the slave data transfer direction.

39.4.2.2.1 7-bit Addressing

The first seven bits of the first byte make up the slave address. The eighth bit is the LSB (least significant bit). It determines the direction of the message. A ‘0’ in the least significant position of the first byte means that the master will write information to a selected slave. A ‘1’ in this position means that the master will read information from the slave.

When an address is sent, each device in a system compares the first seven bits after the START condition with its address. If they match, the device considers itself addressed by the master as a slave-receiver or slave-transmitter, depending on the R/$\overline{W}$ bit.

Each slave must have a unique address. An I²C master must not transmit an address that is the same as its slave address; it cannot be master and slave at the same time.

The slave whose address matches with the address sent by the master pulls SDAH low at the ninth clock (D) to return an acknowledge bit.

39.4.2.3 Data Transfer

When successful slave addressing is achieved, the data transfer can proceed (E) on a byte-by-byte basis in the direction specified by the R/$\overline{W}$ bit sent by the calling master.

Data can be changed only while SCLH is low and must be held stable while SCLH is high, as shown in Figure 39-17. SCLH is pulsed once for each data bit, with the MSB being sent first. The receiving device

must acknowledge each byte by pulling SDAH low at the ninth clock; therefore, a data byte transfer takes nine clock pulses.

If the slave device does not acknowledge the master, it must leave SDAH high. The master can then generate a STOP signal to abort the data transfer or generate a START signal (a repeated start, as shown in Figure 39-18) to start a new calling sequence.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means end-of-data to the slave. The slave releases SDAH for the master to generate a STOP or START signal.

39.4.2.4 STOP Signal

The master can terminate communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDAH while SCLH is at logical high (F).

NOTE

A master can generate a STOP even if the slave has made an acknowledgment; at that point, the slave must release the bus.

39.4.2.5 Repeated Start

Instead of signalling a STOP, the master can repeat the START signal, followed by a calling command (Refer to A in Figure 39-18). A repeated START occurs when a START signal is generated without first generating a STOP signal to end the communication. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

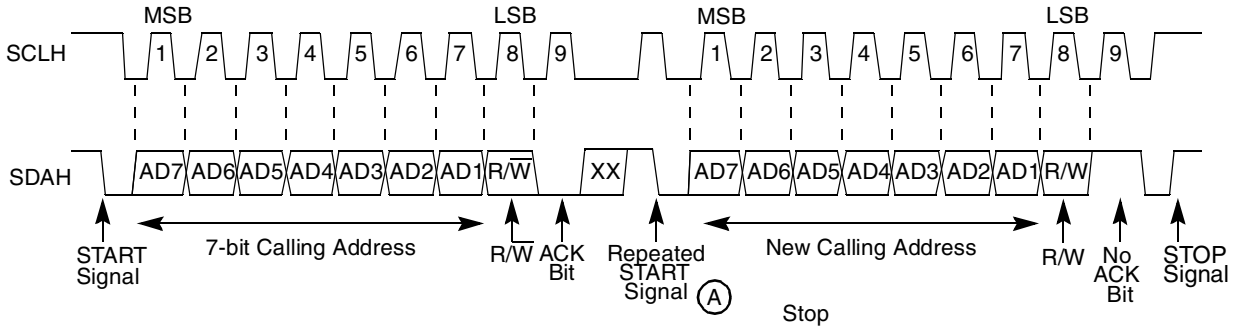


Figure 39-18. Repeated START

39.4.3 Arbitration Procedure

If multiple devices simultaneously request the bus, the bus clock is determined by a synchronization procedure in which the low period equals the longest clock-low period among the devices, and the high period equals the shortest. A data arbitration procedure determines the relative priority of competing devices. A device loses arbitration if it sends logic high while another sends logic low; it immediately switches to slave-receive mode and stops driving SDAH. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets the arbitration lost bit in the HS_I^{2}C Status register (HISR[HIAL] to indicate loss of arbitration.

39.4.4 Clock Synchronization

As wire-AND logic is used, a high-to-low transition on SCLH affects devices connected to the bus. Devices start counting their low period when the master drives SCLH low. When a device clock goes low, it holds SCLH low until the clock high state is reached. However, the low-to-high change in this device clock may not change the state of SCLH if another device clock is still in its low period. Therefore, the device with the longest low period holds the synchronized clock SCLH low. Devices with shorter low periods enter a high wait state during this time (refer to Figure 39-19). When all devices involved have counted off their low period, the synchronized clock SCLH is released and pulled high. There is then no difference between device clocks and the state of SCLH, so all of the devices start counting their high periods. The first device to complete its high period pulls SCLH low again.

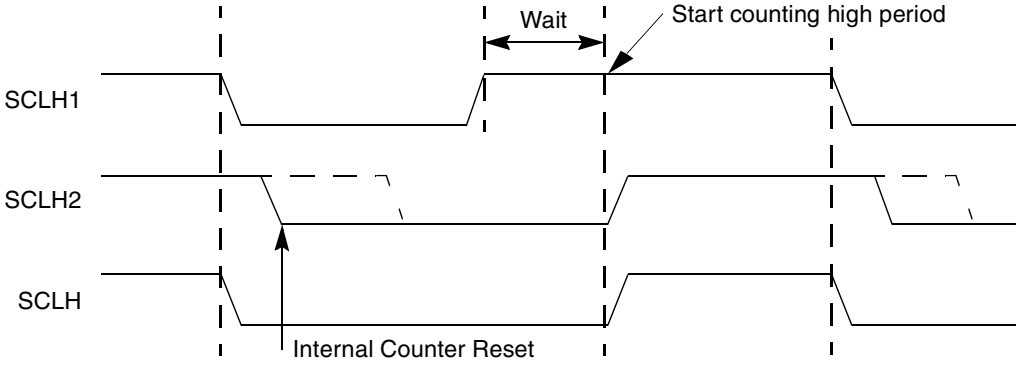


Figure 39-19. Synchronized Clock SCLH

39.4.5 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. Slave devices can hold SCLH low after completing one byte transfer (9 bits). In such a case, the clock mechanism halts the bus clock and forces the master clock into a wait state until the slave releases SCLH.

39.4.6 Clock Stretching

Slaves can use the clock synchronization mechanism to slow down the transfer bit rate. After the master has driven SCLH low, the slave can drive SCLH low for the required period and then release it. If the slave SCLH low period is longer than the master SCLH low period, the resulting SCLH bus signal low period is stretched according to slave SCLH low period.

39.4.7 High-Speed Mode Operation

The HS_I^{2}C module can transfer information at bit rates of up to 3.4 Mbits/s in HS-mode, yet it remains fully downward compatible with Fast or Standard-mode (F/S-mode) devices for bi-directional communication in a mixed-speed bus system. With the exception that arbitration and clock synchronization is not performed during the HS-mode transfer, the same serial bus protocol and data format is maintained as with the F/S-mode system.

Serial data transfer format in HS-mode meets the Standard-mode I^2C-bus specification. HS-mode can only commence after the following conditions are met (all of which occur while in F/S-mode):

1. START condition (S)
2. 8-bit master code (00001XXX)
3. not-acknowledge bit ($\overline{A}$)

Figure 39-20 and Figure 39-21 show this in more detail. The HS master code has two main functions:

1. It allows arbitration and synchronization between competing masters at F/S-mode speeds, resulting in one winning master.
2. It indicates the beginning of an HS-mode transfer.

HS-mode master codes are reserved 8-bit codes, which are not used for slave addressing or other purposes. Furthermore, as each master has its own unique master code, up to eight HS-mode masters can be present on the one I^2C-bus system (although master code 0000 1000 should be reserved for test and diagnostic purposes). The master code for an HS_I^2C module is software programmable in HICR[11-9]. Arbitration and clock synchronization only take place during the transmission of the master code and not-acknowledge bit ($\overline{A}$), after which one winning master remains active. The master code indicates to other devices that an HS-mode transfer is to begin and the connected devices must meet the HS-mode specification. As no device is allowed to acknowledge the master code, the master code is followed by a not-acknowledge ($\overline{A}$). After the not-acknowledge bit ($\overline{A}$), and the SCLH line has been pulled-up to a HIGH level, the active master switches to HS-mode and enables (at time tH, refer to Figure 39-21) the current-source pull-up circuit for the SCLH signal. As other devices can delay the serial transfer before tH by stretching the LOW period of the SCLH signal, the active master will enable its current-source pull-up circuit when all devices have released the SCLH line and the SCLH signal has reached a HIGH level, thus speeding up the last part of the rise time of the SCLH signal. The active master then sends a repeated START condition (Sr) followed by a 7-bit slave address (or 10-bit slave address, refer to Section 39.4.8, 10-Bit Addressing") with a R/$\overline{W}$ bit address, and receives an acknowledge bit (A) from the selected slave.

After a repeated START condition and after each acknowledge bit (A) or not-acknowledge bit ($\overline{A}$), the active master disables its current-source pull-up circuit. This enables other devices to delay the serial transfer by stretching the LOW period of the SCLH signal. The active master re-enables its current-source pull-up circuit again when all devices have released and the SCLH signal reaches a HIGH level, and so speeds up the last part of the SCLH signal s rise time.

Data transfer continues in HS-mode after the next repeated START (Sr), and only switches back to F/S-mode after a STOP condition (P). To reduce the overhead of the master code, it s possible that a master links a number of HS-mode transfers, separated by repeated START conditions (Sr).

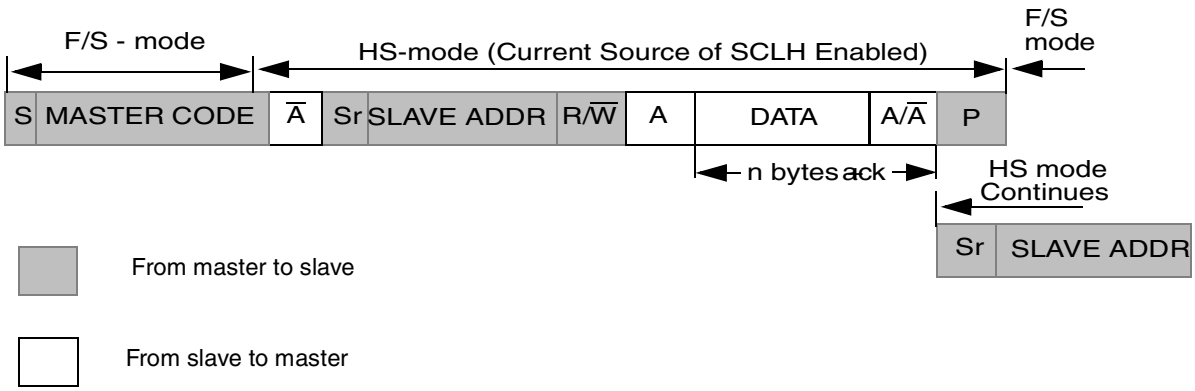


Figure 39-20. Data Transfer Format in HS-Mode

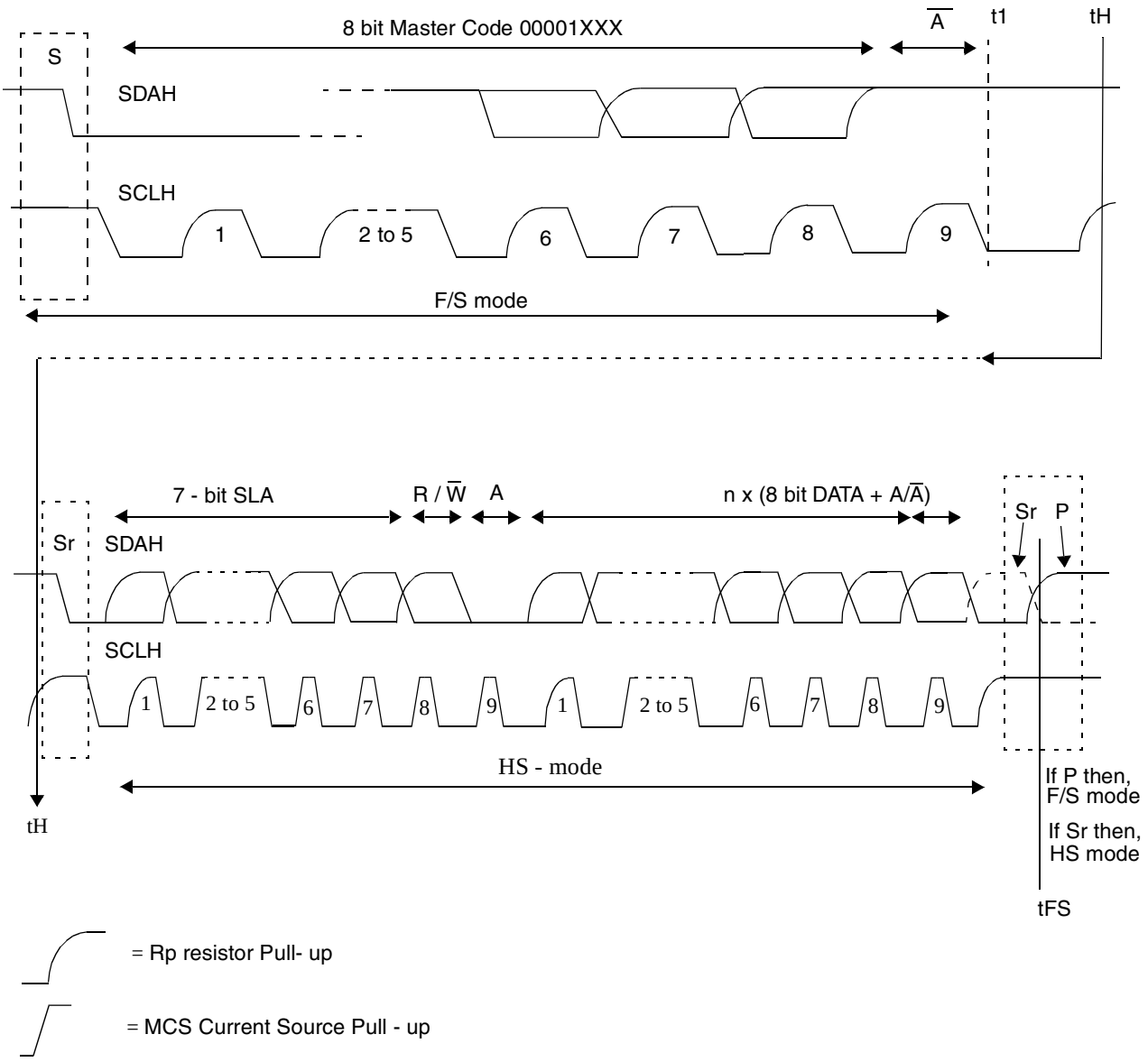


Figure 39-21. A Complete HS-Mode Transfer

39.4.8 10-Bit Addressing

The 10 bit addressing scheme, exploits the reserved combination of 1111XXX. Although there are eight possible combinations of the reserved address bits 1111XXX, only four combinations 11110XX are used for 10-bit addressing. The remaining four combinations of 11111XX are reserved for future I^2C-bus enhancements.

The 10-bit slave address is formed from the first two bytes following a START condition (S) or a repeated START condition (Sr). The first seven bits of the first byte are the combination 11110XX of which the last two bits (XX) are the two most-significant bits (MSBs) of the 10-bit address; the eighth bit of the first byte is the R/$\overline{W}$ bit that determines the direction of the message. A zero in the least significant position of the first byte means that the master will write information to a selected slave. A one in this position means that the master will read information from the slave. If the R/$\overline{W}$ bit is zero, then the second byte contains the remaining 8 bits (XXXXXXXX) of the 10-bit address. If the R/$\overline{W}$ bit is one, then the next byte contains data transmitted from a slave to a master.

Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing. Possible data transfer formats are:

39.4.8.1 Master-Transmitter Transmits to Slave-receiver With a 10-Bit Slave Address.

The Figure 39-22 shows a master transmitter transmits to slave receiver using 10 bit addressing. When a 10-bit address follows a START condition, each slave compares the first seven bits of the first byte of the slave address (11110XX) with its own address and tests if the eighth bit (R/$\overline{W}$ direction bit) is 0. It is possible that more than one device will find a match and generate an acknowledge (A1). All slaves that found a match will compare the eight bits of the second byte of the slave address (XXXXXXXX) with their own addresses, but only one slave will find a match and generate an acknowledge (A2). The matching slave will remain addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

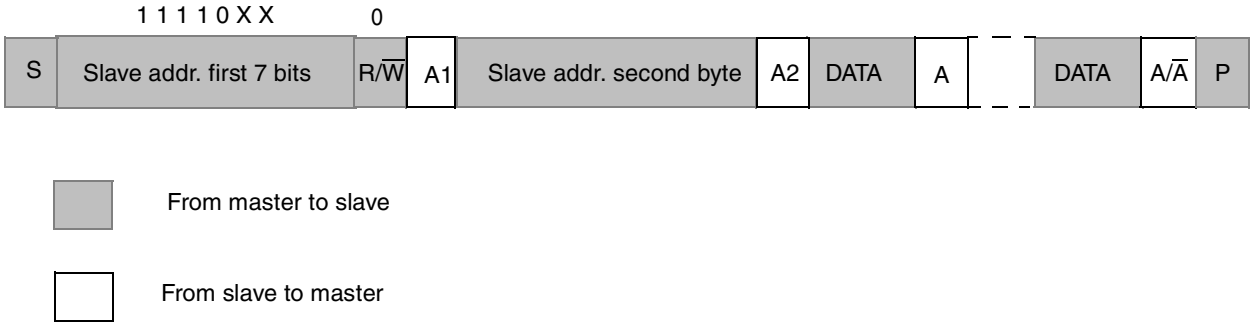


Figure 39-22. Master Transmitter Addresses a Slave Receiver with 10-Bit Address.

39.4.8.2 Master-Receiver Reads Slave-Transmitter With a 10-Bit Slave Address.

The transfer direction is changed after the second R/$\overline{W}$ bit (Refer to Figure 39-23). Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks if the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and tests if the eighth (R/$\overline{W}$) bit is 1. If there is a

match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or until it receives another repeated START condition (Sr) followed by a different slave address. After a repeated START condition (Sr), all the other slave devices will also compare the first seven bits of the first byte of the slave address (11110XX) with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them will be addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

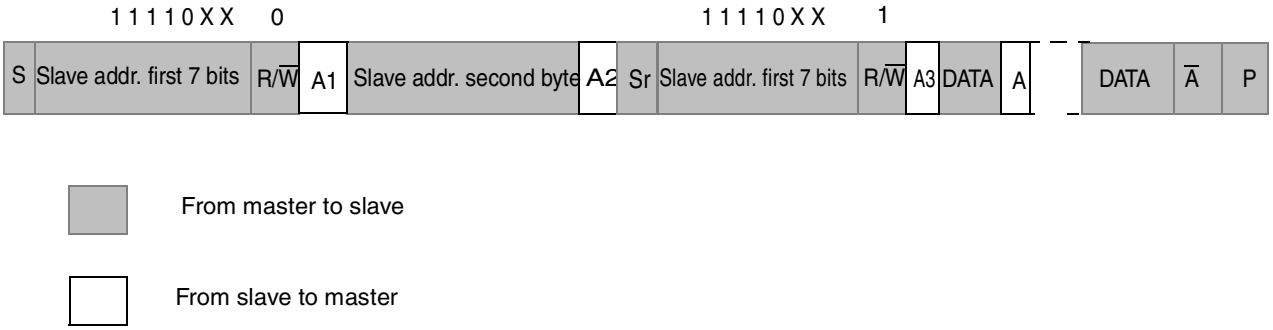


Figure 39-23. A Master-receiver Addresses a Slave-Transmitter With a 10-bit Address.

39.4.8.3 Combined Format

The combined formats like, a master transmits data to one slave and then transmits data to another slave is also possible. Another combined format where 10-bit and 7-bit addressing combined in one serial transfer is also possible. In this case the same master occupies the bus all the time.

39.4.9 DMA Bus Interface

The IP bus interface is used for both HS_I^{2}C registers read/write operation as well as DMA transfer. The core can program the HS_I^{2}C to transfer data to or from memory through the DMA controller. The interface uses two DMA channels: one to write transmit data to the HS_I^{2}C, and one to read the received data from the HS_I^{2}C module. In each case, a transfer is initiated by a dedicated DMA receive or transmit request from the HS_I^{2}C. Figure 39-24 shows the DMA interface operation of HS_I^{2}C.

The DMA will be used for data transfer operation only. The address register, HIMADR is used for slave address transfer in master mode and HISADR is used for slave address comparison for slave mode. If the DMA and FIFO are enabled, and HS_I^{2}C is in master mode and looses bus arbitration, software needs to re-configure the module for master mode operation if the module needs to be operated in master mode.

When HS_I^{2}C is configured for master transmit operation, DMA and FIFO are enabled and for a particular data or address cycle ACK is not received and HICR[AUTO_RSTA] bit is set, HS_I^{2}C will generate the repeated operation and mean time interrupt will be generated to indicate the ACK has not been received. After the address phase the same data will be transmitted again for which ACK is not received so that there is no loss of data.

39.4.9.1 DMA Transmit Request

If the TX FIFO is enabled and DMA_EN_TX is set, the DMA transmit request is generated when the data level in the TX FIFO is less than or equal to the programmable Transmit FIFO Empty WaterMark (TFWM[4:0]).

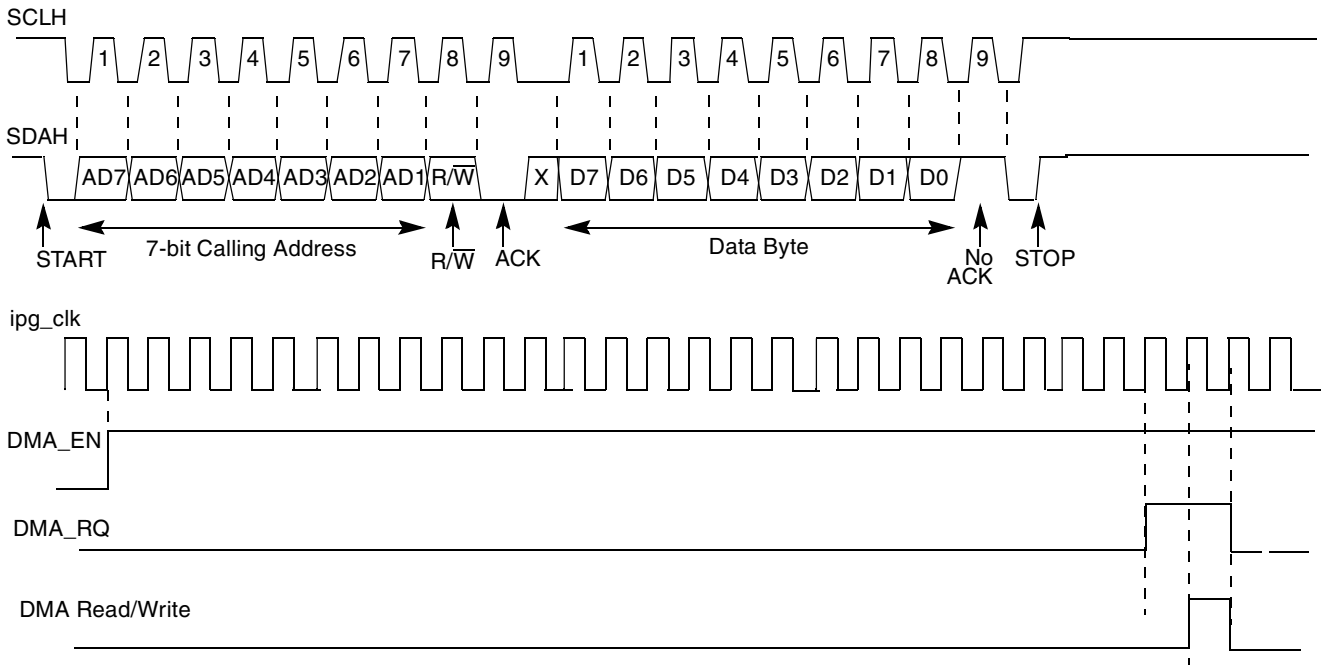


Figure 39-24. DMA Interface Operation of HS_I^{2}C

39.4.9.2 DMA Receive Request

If the RX FIFO is enabled and DMA_EN_RX is set, the DMA receive request is generated when the data level in the RX FIFO is greater than or equal to the programmable Receive FIFO Full WaterMark (RFWM[5:0]).

39.4.10 Receive and Transmit FIFOs

A 8 × 8 Tx FIFO and a 8 × 8 Rx FIFO are present to buffer HS_I^{2}C Tx and Rx data respectively. Figure 39-25 shows a high level block diagram of FIFOs and associated control registers in the HS_I^{2}C.

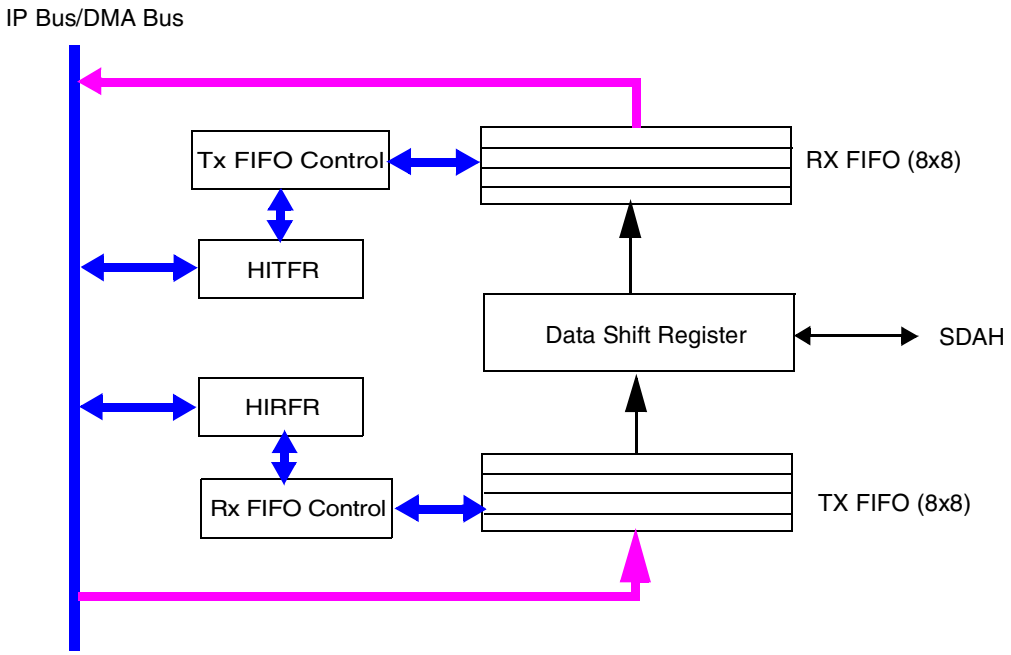


Figure 39-25. Block Diagram of FIFOs and its Control Registers in HS_I^{2}C

39.4.11 FIFO Flush Operation

The RX and TX FIFOs are reset by a Hardware or software reset. To reset only the FIFOs without affecting the HS_I^{2}C data transfer configuration, there are control bits provided in the HITFR and HIRFR registers for Tx and Rx FIFO respectively. The RFLSH bit is used to reset the RX FIFO and TFLSH bit is used to reset the TX FIFO. This RX FIFO/ TX FIFO reset using these control bits is referred to as RX FIFO/ TX FIFO flush operation. The ipg_hsi2c_clk should be ON for Tx/Rx FIFO flush operation.

39.4.12 IP Bus Accesses

HS_I^{2}C is a 32-bit IP module. Only 16 bit and 8 bit accesses should be performed to the module.

39.4.13 Generation of Transfer Error on IP Bus

If an address is received on the IP slave bus interface that is not implemented, an access error is generated (ips_xfr_err is asserted). The input pin resp_sel provides the configuration capability to generate this response. The resp_sel pin must be asserted to enable the ips_xfr_err signal.

39.4.14 Clocks

There are two clock sources for HS_I^{2}C module as follows.

- ipg_clk_s—The clock is used for IP bus register read/writes.
- ipg_hsi2c_clk—This is the functional clock of the HS_I^{2}C module. The serial bit clock frequency is derived from the ipg_hsi2c_clk. The ipg_hsi2c_clk and ipg_clk_s clocks are asynchronous to each other. The minimum frequency of this clock should be 66.66 MHz for HS-mode operation.

39.4.15 Reset

The HS_I^{2}C module can be reset in two ways.

- Global reset: It is a global reset (ipg_hard_async_reset_b) and resets the whole HS_I^{2}C module.
- Software reset: The HICR[HIEN] bit when de-asserted will generate an internal reset which will reset part of the module.

39.4.16 Interrupts

There is only one interrupt line from the module. Interrupt is enabled by setting the HICR[HIIEN] bit. The interrupt is generated in any one of the following conditions.

- One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock).
- An address is received that matches its own specific address in slave-receive mode.
- Arbitration is lost.
- Tx FIFO Transmit data empty
- Rx FIFO Receive data full

There are separate interrupt mask bits in HIIMR register for each individual interrupt condition.

39.4.17 Endianness

The module only supports the little endian mode.

39.5 Initialization Information

The HS_I^{2}C needs to be initialized before the data transfer can take place. The initialization sequence for DMA and FIFO disabled condition is as follows:

1. Set the data sampling rate (HIFSFDR[FSICR]) to obtain SCLH frequency from the ipg_hsi2c_clk clock for F/S-Mode operation. Program the HIHSFDR[HSICR] to obtain the HS-Mode operating frequency.
2. Update the address in the (HISADR and HIMADR) to define its slave and master address for both 7 and 10 bit addressing.
3. Set the I^{2}C enable bit (HICR[HIEN]) to enable the HS_I^{2}C bus interface system.
4. Modify the bits in the HICR to select master/slave mode, transmit/receive mode, high-speed or F/S mode, 7 or 10 bit addressing and interrupt-enable or not.

39.5.1 Generation of START

After completion of the initialization procedure, serial data can be transmitted by selecting the master transmitter mode. In multi-master bus system, the busy bus (HISR[HIBB]) must be tested to determine whether the serial bus is free. If the bus is free (HIBB = 0), the START signal and the first byte (the slave address for 7-bit addressing or 10-bit address code for 10 bit addressing or HS mode code) can be sent. The data written to the data register comprises the address of the desired slave and the LSB indicates the transfer direction in case of 7-bit addressing mode. For 10-bit addressing mode, the first byte contains the 10 bit address code and the MSB two bits of the 10 bit address. The second byte contains the remaining 8 bits of the 10 bit address. Refer to Section 39.4.8, 10-Bit Addressing for more details on 10-bit addressing mode. The status bit HISR[SADDR_MODE] indicates whether 7/10-bit addressing mode is used when HS_I2C is selected as slave.

For High-Speed mode, the first byte after start/repeated start condition is 8 bit master code (the value in the HICR[HS_MST_CODE] for master mode). When HS_I^2C is configured as slave, the status bit HISR[SHS_MODE] indicates whether the module is selected for HS or F/S mode.

39.5.2 Post-Transfer Software Response

Sending or receiving a byte sets the byte transfer done bit (HISR[BTD]), which indicates one byte communication is finished. Upon data transfer completion and if HIIMR[MASK_BTD] is cleared and interrupt enable (HICR[HIIEN]) is set, an external interrupt is generated. The byte transfer done bit (HISR[BTD]) is cleared by writing 1 to this bit.

During slave-mode address cycles (HISR[HIAAS] = 1), the slave read/write bit HISR[SRW] is read to determine the direction of the next transfer. For slave-mode data cycles (HISR[HIAAS] = 0), HISR[SRW] is invalid.

39.5.3 Generation of STOP

A data transfer ends when the master signals a STOP, which can occur after all data is sent.

For a master receiver to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last data byte. This is done by setting the transmit acknowledge bit (HICR[TXAK]) before reading the next-to-last byte. Before the last byte is read, a STOP signal must be generated.

39.5.4 Generation of Repeated START

After the data transfer, if the master still wants the bus, it can signal another START followed by another slave address without signalling a STOP. Repeated start can be programmed by setting HICR[RSTA] bit.

39.5.5 Slave Mode

In the slave interrupt service routine (Refer to flow chart Figure 39-26), the module addressed as slave bit (HIAAS) should be tested to check if a calling of its own address has just been received. The only time HIAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred.

To clear the HISR[HIAAS], software should write 1 to this bit. A data transfer can now be initiated by writing information to HITDR for slave transmits, or read from HIRDR in slave-receive mode.

In the slave transmitter routine, the receive acknowledge bit (HISR[RXAK]) must be tested before sending the next byte of data. Setting of HISR[RXAK] means an end-of-data signal from the master receiver.

39.5.6 Arbitration Lost

If several devices try to engage the bus at the same time, one becomes master. Hardware immediately switches devices that lose arbitration to slave receive mode. Data output to SDAH stops, but SCLH is still generated until the end of the byte during which arbitration is lost. An interrupt occurs if the arbitration is lost (HISR[HIAL] = 1), and the slave mode is selected (HICR[MSTA] = 0). Refer to flow chart in Figure 39-26.

If the HS_I2C module is not a master and tries to transmit or does a START, hardware inhibits the transmission, clears MSTA without signalling a STOP, generates an interrupt to the CPU, and sets HISR[HIAL] to indicate a failed attempt to engage the bus. When considering these cases, the slave service routine should first test HISR[HIAL], and the software should clear it if it is set.

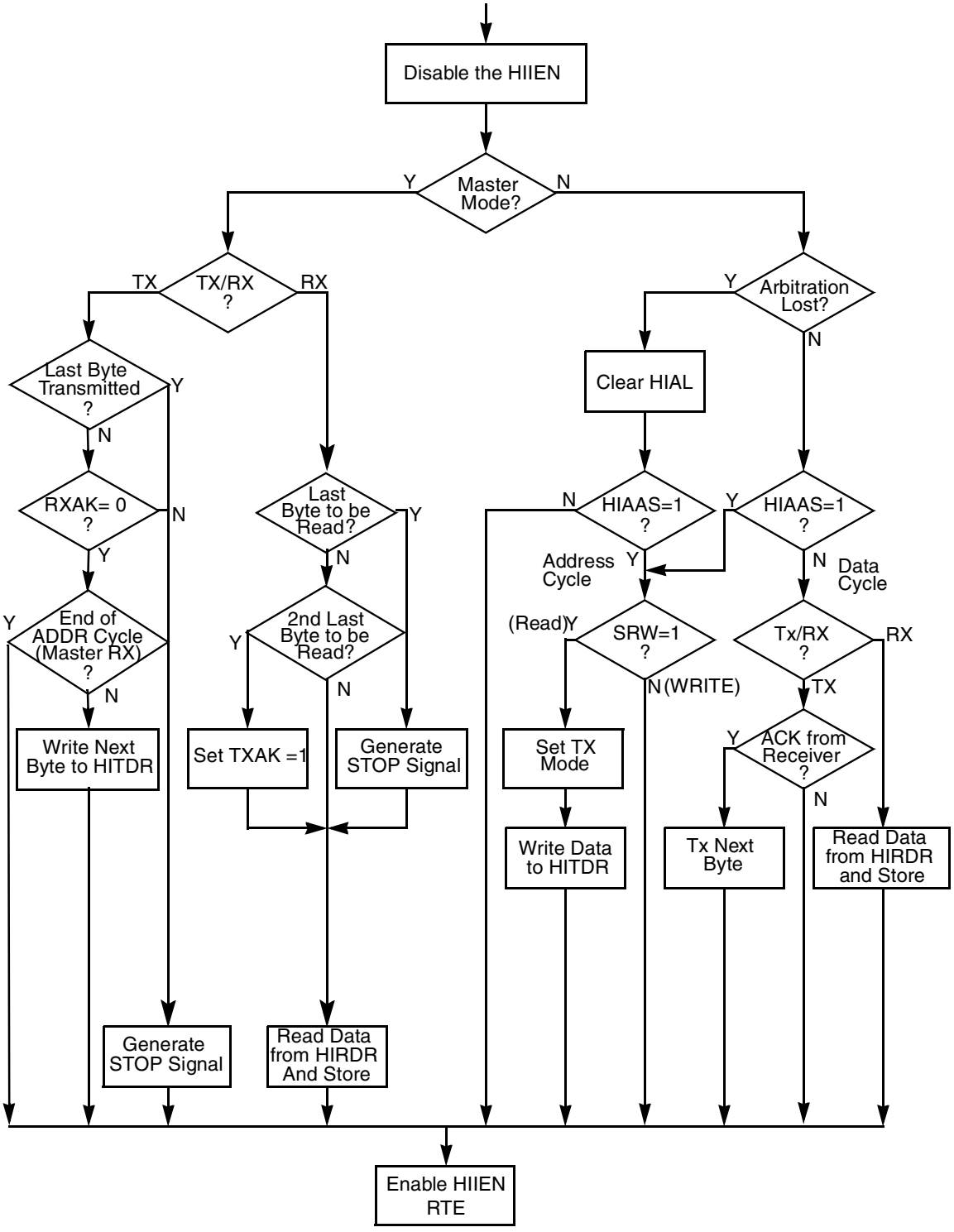


Figure 39-26. Flow-Chart of Typical HS_I^{2}C Interrupt Routine when Tx/Rx FIFO disabled

39.6 Application Information

39.6.1 Programming Sequence:

39.6.1.1 7Bit/10Bit Address Master Transmit:

39.6.1.1.1 DMA_EN = 0 and FIFO_EN = 0

1. Program HIFSFDR[FSICR] register to set SCLH frequency for F/S-Mode operation. Similarly program HIHSFDR[HSICR] register to set HS-Mode operating frequency.
2. Enable HS_I^{2}C by setting HICR[HIEN] to 1.
3. Program the HICR[ADDR_MODE] bit for selecting between 7 bit and 10 bit addressing mode. Program the LSB/MSB address of the slave to be transmitted to HIMADR.
4. Program the HICR[HS_MST_CODE] in case of High Speed transfer. Set the HICR[MTX] bit to indicate transmit operation.
5. Enable HS_I^{2}C interrupt by setting HICR[HIIEN] bit and the HIIMR[MSK_BTD] bit to interrupt the core when a byte transfer is complete.
6. Write one Byte of DATA to be transmitted to HITDR.
7. Read HISR[HIBB] to bit to make sure I^{2}C bus is free and initiate Master mode transmit/receive by setting HICR[MSTA] bit.
8. Wait until the core is interrupted by HS_I^{2}C interrupt when BTD is set for the second time at which time the first data would have been transmitted.
9. Program the next byte of data to be transmitted in HITDR. Clear the HS_I^{2}C interrupt by writing 1 to HISR[BTD].
10. Continue steps 8-9 until transmit is required to that Slave.
11. In order to initiate a repeated start, program the next slave address to HIMADR register. Then set HICR[RSTA] bit to initiate the repeated start at the end of current byte transfer.
12. Complete the transmit by Clearing the HICR[MSTA] which would initiate a STOP condition on I^{2}C bus.

39.6.1.1.2 DMA_EN = 1 and FIFO_EN = 1

Follow Steps 1 to 5, similar to DMA=0 and FIFO_EN=0 condition above and

1. Program the HITFR[TFWM] field with appropriate value for transmit. Set the HITFR[TFEN] bit to enable the TX FIFO. Flush the TX FIFO by setting the HITFR[TFLSH] bit.
2. Program system's DMA for appropriate Data Transmit to HS_I^{2}C. Set the HICR[DMA_EN_TX] bit to enable DMA requests for TX FIFO.
3. Initiate Master mode transmit/receive by setting HICR[MSTA] bit.
4. Data Transmit would be continued as long as there is valid data in the TXFIFO.
5. In order to initiate a repeated start, program the next slave address to HIMADR register. Then set HICR[RSTA] bit to initiate the repeated start at the end of current byte transfer.

6. Master mode transmit can be stopped by Clearing the HICR[MSTA] which would initiate a STOP condition on I^2C bus.
7. The DMA request generation should be disabled by clearing the HICR[DMA_EN_TX] bit.

39.6.1.2 7Bit/10Bit Address Master Receive:

39.6.1.2.1 DMA_EN = 0 and FIFO_EN = 0

1. Program HIFSFDR[FSICR] register to set SCLH frequency for F/S-Mode operation. Similarly program HIHSFDR[HSICR] register to set HS-Mode operating frequency.
2. Enable HS_I^2C by setting HICR[HIEN] to 1.
3. Program the HICR[ADDR_MODE] bit for selecting between 7 bit and 10 bit addressing mode. Program the LSB/MSB address of the slave to receive from, in the HIMADR register.
4. Program the HICR[HS_MST_CODE] in case for High Speed transfer. Set the HICR[MTX] bit to indicate a receive operation. Also program the HICR[TXACK] bit accordingly.
5. Enable HS_I^2C interrupt by setting HICR[HIIEN] bit and the HIIMR[MSK_BTD] bit to interrupt the core when a byte transfer is complete.
6. Initiate the Master mode Receive by setting HICR[MSTA] bit.
7. Wait until the core is interrupted by the HS_I^2C interrupt when BTD is set.
8. Fetch the data received from the slave by reading HIRDR register. Clear the HS_I^2C interrupt by writing 1 to HISR[BTD].
9. Continue steps 7-8 until receive operation is required to be continued.
10. In order to initiate a repeated start, program the next slave address to be addressed in the HIMADR register. Then set HICR[RSTA] bit to initiate a repeated start at the end of current byte transfer.
11. Complete the transmit by Clearing the HICR[MSTA] which would initiate a STOP condition on I^2C bus.

39.6.1.2.2 DMA_EN = 1 and FIFO_EN = 1

Follow Steps 1 to 5, similar to DMA=0 and FIFO_EN=0 condition above and

1. Program the HIRFR[RFWM] field with appropriate value to interrupt the DMA. Set the HIRFR[RFEN] bit to enable the RX FIFO. Flush the RX FIFO by setting the HIRFR[RFLSH] bit.
2. Program the system's DMA for appropriate Data Receive from HS_I^2C. Set the HICR[DMA_EN_RX] bit to enable DMA requests from HS_I^2C to the system's DMA.
3. Initiate Master mode transmit/receive by setting HICR[MSTA] bit.
4. RX DMA requests would be generated whenever DATA in RX FIFO equals/exceeds RX watermark value.
5. In order to initiate a repeated start, program the next slave address to HIMADR register. Then set HICR[RSTA] bit to initiate the repeated start at the end of current byte transfer.
6. Data Receive would be continued until the master mode receive operation is stopped by Clearing the HICR[MSTA] which would initiate a STOP condition on I^2C bus.

7. The DMA request generation should be disabled by clearing the HICR[DMA_EN_RX] bit.

39.6.1.3 7Bit/10Bit Address Slave Transmit:

39.6.1.3.1 DMA_EN = 0 and FIFO_EN = 0

1. Program HIFSFDR[FSICR] register to set SCLH frequency for F/S-Mode operation. Similarly program HIHSFDR[HSICR] register to set HS-Mode operating frequency.
2. Program the Slave Address to which HS_I^{2}C should respond as slave to HISADR register. Also program the HICR[SAMC] field to indicate the addressing mode to which HS_I^{2}C should respond.
3. Enable HS_I^{2}C by setting HICR[HIEN] to 1.
4. Enable HS_I^{2}C interrupt by setting HICR[HIIEN] bit and the HIIMR[HIAAS] bit to interrupt the core whenever HS_I^{2}C is addressed as slave.
5. When HS_I^{2}C interrupt is asserted and HIAAS bit is set, read the HISR[SRW] bit and write the corresponding value to the HICR[MTX] bit.
6. If HS_I^{2}C has been addressed to Transmit, write a Byte of data into HITDR register. Clear the interrupt by writing i to HISR[HIAAS] bit. Enable the BTD interrupt and RXACK interrupt by setting HIIMR[BTD] and HIIMR[RXACK] bits.
7. Wait until the core is interrupted by the HS_I^{2}C interrupt when HISR[BTD] is set.
8. Program the next byte of data to be transmitted in HITDR. Clear the HS_I^{2}C interrupt by writing 1 to HISR[BTD].
9. Continue steps 8-9 until a NOT ACK interrupt is received from the master or a STOP condition is detected which would clear the HISR[HIBB] bit. Clear the HS_I^{2}C interrupt by writing 1 to HISR[RXACK].
10. If repeated Start is detected on the I^{2}C bus, the procedure would follow from step 5.

39.6.1.3.2 DMA_EN = 1 and FIFO_EN = 1

Follow Steps 1 to 4, similar to DMA=0 and FIFO_EN=0 condition above and

1. Program the HITFR[TFWM] field with appropriate value for transmit. Set the HITFR[TFEN] bit to enable the TX FIFO. Flush the TX FIFO by setting the HITFR[TFLSH] bit.
2. Program system's DMA for appropriate Data Transmit to HS_I^{2}C. Set the HICR[DMA_EN_TX] bit to enable DMA requests for TX FIFO.
3. When HS_I^{2}C interrupt is asserted and HIAAS bit is set, read the HISR[SRW] bit and write the corresponding value to the HICR[MTX] bit. Enable the RXACK interrupt by setting HIIMR[RXACK] bits.
4. If HS_I^{2}C has been addressed to Transmit, data Transmit would be continued to the master as long as there is valid data in the TXFIFO or until a NOT ACK interrupt is received from the master or a STOP condition is detected which would clear the HISR[HIBB] bit. Clear the HS_I^{2}C interrupt by writing 1 to HISR[RXACK].
5. If repeated Start is detected on the I^{2}C bus, the procedure would follow from step 3.
6. The DMA request generation should be disabled by clearing the HICR[DMA_EN_TX] bit.

39.6.2 Programming Restrictions

Programming restrictions are as follows:

1. The HIFSFDR register and HIHSFDR register can be programmed only when HICR[HIEN] is 0.
2. The HISADR register and HICR[SAMC] field can be programmed only when HICR[HIEN] is 0.
3. The HICR[HS_MST_CODE] field can be programmed only when HICR[MSTA] bit is 0.
4. The HIMADR register can be programmed only when
   - HICR[MSTA] bit is 0. After HICR[MSTA] bit is set to 1, HIMADR should not be programmed for 4 cycles of ipg_hsi2c_clk.
   - HICR[RSTA] bit is 0. After HICR[RSTA] is set the HIMADR register should not be programmed until RSTA bit is cleared.
5. All other Control Registers can be programmed only when HICR[HIEN] is 1.
6. The HITDCR[TDC] field can be programmed only when HITDCR[TDC_EN] = 0 and similarly HIRDCR[RDC] field can be programmed only when HIRDCR[RDC_EN] = 0.
7. When changing from FIFO enable to disable mode, FIFO flush should be done.
8. The HITDCR[TDC_RSTA] and HIRDCR[RDC_RSTA] bits repeat start functionality should not be used for switching the configuration from master Tx to master Rx and vice versa. HITDCR[TDC_RSTA] and HIRDCR[RDC_RSTA] bits repeat start functionality is meant for data transfer to different slave devices for Tx only and Rx only cases.

39.6.3 Timing Section

Refer to Electrical Specification document for the HS_I^2C module timing characteristics.

Chapter 40
Inter IC (I^2C)

40.1 Introduction

The Inter IC (I^2C) module provides functionality of a standard I^2C slave and master. The I^2C module is designed to be compatible with the standard Philips I^2C bus protocol. See the I^2C block diagram in Figure 40-1.

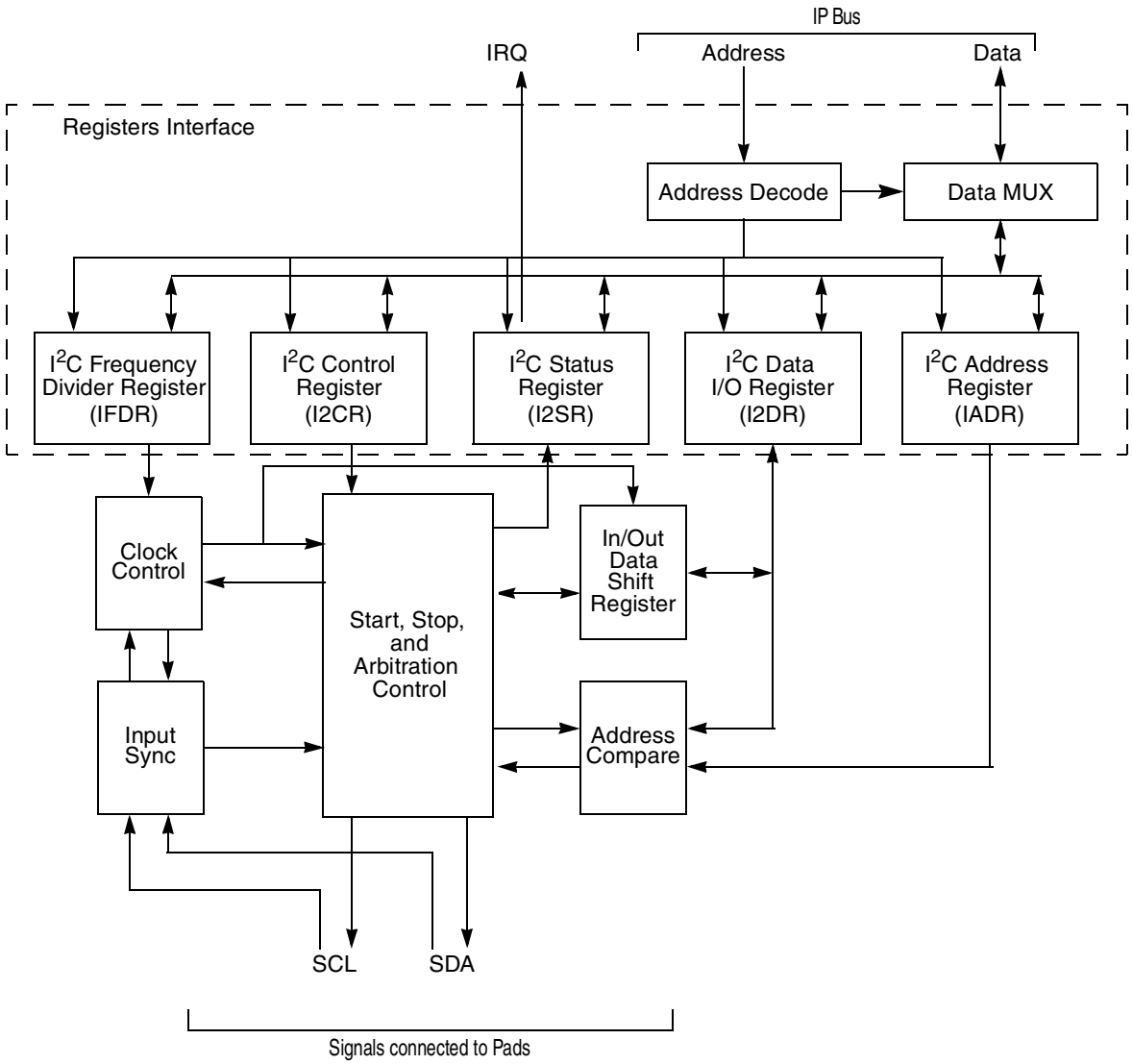


Figure 40-1. I^2C Block Diagram

40.1.1 Overview

I^{2}C is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices. This bus is suitable for applications requiring occasional communications over a short distance between many devices. The flexible I^{2}C allows additional devices to be connected to the bus for expansion and system development. See the connection diagram in Figure 40-2.

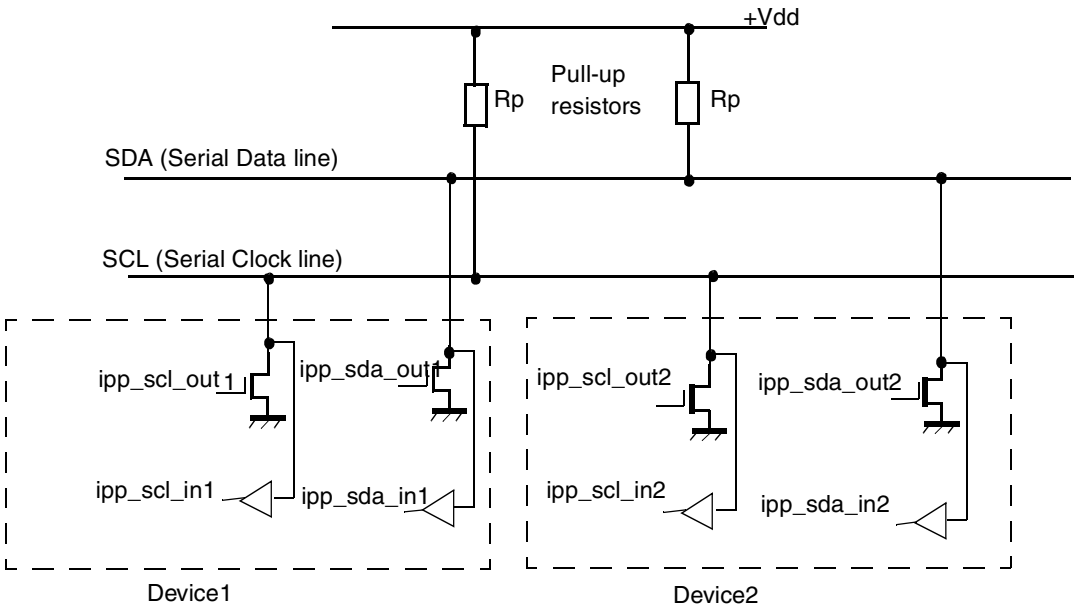


Figure 40-2. Connection of Devices to I^{2}C Bus

The I^{2}C operates up to 400 kbps, but it depends on the pad loading and timing. For pad requirement details, refer to Philips I^{2}C Bus Specification, Version 2.1. The I^{2}C system is a true multiple-master bus including arbitration and collision detection that prevents data corruption if multiple devices attempt to control the bus simultaneously. This feature supports complex applications with multiprocessor control and can be used for rapid testing and alignment of end products through external connections to an assembly-line computer.

40.1.2 Features

The I^{2}C module has the following key features:

- Compatibility with I^{2}C bus standard
- Multiple-master operation
- Software-programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven, byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt

- Start and stop signal generation/detection
- Repeated START signal generation
- Acknowledge bit generation/detection
- Bus-busy detection

40.2 External Signal Description

Two pins are required for I^2C:

- I2C_SCL Bidirectional Clock Pin
- I2C_SDA Bidirectional Data Pin

For I^2C compliance, all devices connected to the SCL and SDA signals must have open-drain or open-collector outputs. The logic AND function is exercised on both lines with external pull-up resistors.

The module port signals going to the pad are tabulated in Table 40-2:

Table 40-2. Signal Properties

Name	Port	Function	Reset State	Pull up
ipp_scl_in	—	Serial Clock input	1	—
ipp_scl_out	—	Serial Clock output	1	Active
ipp_scl_out_en	—	Serial Clock output enable	0	—
ipp_sda_in	—	Serial Data input	1	—
ipp_sda_out	—	Serial Data output	1	Active
ipp_sda_out_en	—	Serial Data output enable	0	—

40.2.1 Detailed External Signal Descriptions

The following three signals are connected to the bidirectional driver for the SCL I/O Pad (through the IOMUX module):

- ipp_scl_in—Serial Input Clock
- ipp_scl_out—Serial Output Clock
- ipp_scl_out_en—Serial Output Clock Enable

The pad needs to have open-drain connectivity.
The ipp_scl_in will be the input signal from the pad. The ipp_scl_out will be the output to the pad from the module. The ipp_scl_out_en will act as the output enable.

The following three signals are connected to the bidirectional driver for the SDA I/O Pad (through the IOMUX module):

- ipp_sda_in—Serial Input Data
- ipp_sda_out—Serial Output Data
- ipp_sda_out_en—Serial Output Data Enable

The pad should have open-drain connectivity. The ipp_sda_in will be the input signal from the pad. The ipp_sda_out will be the output to the pad from module. The ipp_sda_out_en will act as the output enable.

40.3 Memory Map and Register Definition

The I^{2}C module contains five 16-bit registers. Section 40.3.3, Register Descriptions" on page 40-6 provides the detailed descriptions for all of the I^{2}C registers.

40.3.1 I^{2}C Memory Map

Table 40-3 shows the I^{2}C memory map.

Table 40-3. I^{2}C Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x000(IADR)	I^{2}C Address Register	R/W	0x0000	40.3.3.1/40-6
0xBASE+0x004(IFDR)	I^{2}C Frequency Divider Register	R/W	0x0000	40.3.3.2/40-6
0xBASE+0x008(I2CR)	I^{2}C Control Register	R/W	0x0000	40.3.3.3/40-8
0xBASE+0x00C(I2SR)	I^{2}C Status Register	R/W	0x0081	40.3.3.4/40-9
0xBASE+0x010(I2DR)	I^{2}C Data I/O Register	R/W	0x0000	40.3.3.5/40-10

NOTE

Registers at addresses 0xBASE_02, 0xBASE_06, 0xBASE_a, 0xBASE_0e are reserved for future additions.

40.3.2 Register Summary

Figure 40-3 shows the key to the register fields and Table 40-5 shows the register figure conventions.

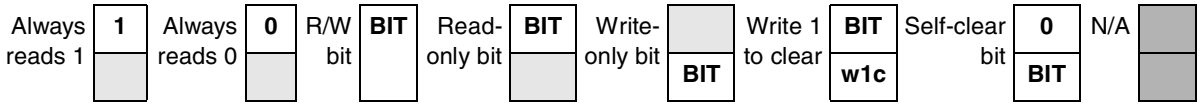


Figure 40-3. Key to Register Fields

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 40-5. Register Figure Conventions

Table 40-4 shows the I^{2}C register summary.

Table 40-4. I^{2}C Register Summary

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000(IADR)	R	0	0	0	0	0	0	0	0	ADR							0
	W																
0xBASE+0x004(IFDR)	R	0	0	0	0	0	0	0	0	0	0	IC					
	W																
0xBASE+0x008(I2CR)	R	0	0	0	0	0	0	0	0	IEN	IIEN	MSTA	MTX	TXAK	0	0	0
	W														RSTA		
0xBASE+0x00C(I2SR)	R	0	0	0	0	0	0	0	0	ICF	IAAS	IBB	IAL	0	SRW	IIF	RXAK
	W																
0xBASE+0x010(I2DR)	R	0	0	0	0	0	0	0	0	DATA							
	W																

40.3.3 Register Descriptions

This section contains the detailed register descriptions for the I^{2}C registers in address order.

40.3.3.1 I^{2}C Address Register (IADR)

Figure 40-6 shows the I^{2}C Address Register; Table 40-5 provides its field descriptions.

0xBASE+0x000(IADR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	ADR							0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 40-6. I^{2}C Address Register

The IADR holds the address the I^{2}C responds to when addressed as a slave.

NOTE

The slave address is not the address sent on the bus during the address transfer. The register is not reset by a software reset.

Table 40-5. I^{2}C Address Register Field Descriptions

Field	Description
15–8	Reserved
7–1 ADR	Slave address. Contains the specific slave address to be used by the I^{2}C module. Slave mode is the default I^{2}C mode for an address match on the bus.
0	Reserved

40.3.3.2 I^{2}C Frequency Register (IFDR)

The IFDR provides a programmable prescaler to configure the clock for bit-rate selection. The register does not get reset by software reset. Figure 40-7 shows the I^{2}C frequency register; Table 40-6 provides its field descriptions.

0xBASE+0x004(IFDR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	IC					
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 40-7. I^{2}C Frequency Register

Table 40-6. I²C Frequency Register Field Descriptions

Field	Description
15–6	Reserved
5–0 IC	I²C clock rate. Pre scales the clock for bit-rate selection. Due to potentially slow SCL and SDA rise and fall times, bus signals are sampled at the prescaler frequency. The serial bit clock frequency is equal to IPG_CLK_PATREF divided by the divider shown in Table 40-7. **Note:** The IC can be changed anywhere in a program. I²C protocol supports bit rates up to 400 kbps. The IC bits need to be programmed in accordance with this constraint.

Table 40-7. IFDR Register Field Values

IC	Divider	IC	Divider	IC	Divider	IC	Divider
0x00	30	0x10	288	0x20	22	0x30	160
0x01	32	0x11	320	0x21	24	0x31	192
0x02	36	0x12	384	0x22	26	0x32	224
0x03	42	0x13	480	0x23	28	0x33	256
0x04	48	0x14	576	0x24	32	0x34	320
0x05	52	0x15	640	0x25	36	0x35	384
0x06	60	0x16	768	0x26	40	0x36	448
0x07	72	0x17	960	0x27	44	0x37	512
0x08	80	0x18	1152	0x28	48	0x38	640
0x09	88	0x19	1280	0x29	56	0x39	768
0x0A	104	0x1A	1536	0x2A	64	0x3A	896
0x0B	128	0x1B	1920	0x2B	72	0x3B	1024
0x0C	144	0x1C	2304	0x2C	80	0x3C	1280
0x0D	160	0x1D	2560	0x2D	96	0x3D	1536
0x0E	192	0x1E	3072	0x2E	112	0x3E	1792
0x0F	240	0x1F	3840	0x2F	128	0x3F	2048

40.3.3.3 I²C Control Register (I2CR)

The I2CR is used to enable the I²C module and the I²C interrupt. It also contains bits that govern operation as a slave or a master. Figure 40-8 shows the I²C Control Register; Table 40-8 provides its field descriptions.

0xBASE+0x008(I2CR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	IEN	IIEN	MSTA	MTX	TXAK	0	0	0
W														RSTA		
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 40-8. I²C Control Register

Table 40-8. I²C Control Register Field Descriptions

Field	Description
15–8	Reserved
7 IEN	I²C enable. Also controls the software reset of the entire I²C module. Resetting the bit generates an internal reset to the module. If the module is enabled in the middle of a byte transfer, slave mode ignores the current bus transfer and starts operating when the next start condition is detected. Master mode is not aware that the bus is busy; so initiating a start cycle may corrupt the current bus cycle, ultimately causing either the current master or the I²C module to lose arbitration. After which, bus operation returns to normal. 0 The module is disabled, but registers can still be accessed. 1 The I²C module is enabled. This bit must be set before any other I2CR bits have any effect.
6 IIEN	I²C interrupt enable. 0 I²C module interrupts are disabled, but the status flag I2SR[IIF] continues to be set when an interrupt condition occurs. 1 I²C module interrupts are enabled. An I2C interrupt occurs if I2SR[IIF] is also set.
5 MSTA	Master/slave mode select bit. If the master loses arbitration, MSTA is cleared without generating a STOP signal. **Note:** Module clock should be on for writing to the MSTA bit. 0 Slave mode. Changing MSTA from 1 to 0 generates a STOP and selects slave mode. 1 Master mode. Changing MSTA from 0 to 1 signals a START on the bus and selects master mode.
4 MTX	Transmit/receive mode select bit. Selects the direction of master and slave transfers. 0 Receive. When a slave is addressed, the software should set MTX according to the slave read/write bit in the I²C status register (I2SR[SRW]). 1 Transmit. In master mode, MTX should be set according to the type of transfer required. Therefore, for address cycles, MTX is always 1.
3 TXAK	Transmit acknowledge enable. Specifies the value driven onto SDA during acknowledge cycles for both master and slave receivers. **Note:** Writing TXAK applies only when the I²C bus is a receiver. 0 An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte of data. 1 No acknowledge signal response is sent (that is, the acknowledge bit = 1).

Table 40-8. I²C Control Register Field Descriptions

Field	Description
2 RSTA	Repeat start. Always reads as 0. Attempting a repeat start without bus mastership causes loss of arbitration. 0 No repeat start 1 Generates a repeated START condition
1–0	Reserved

40.3.3.4 I²C Status Register (I2SR)

The I2SR contains bits that indicate transaction direction and status. Figure 40-9 shows the I²C Address Register; and Table 40-10 provides its field descriptions.

0xBASE+0x00C(I2SR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	ICF	IAAS	IBB	IAL	0	SRW	IIF	RXAK
W																
RESET	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Figure 40-9. I²C Status Register

Table 40-10. I²C Status Register Field Descriptions

Field	Description
15–8	Reserved
7 ICF	Data transferring bit. While one byte of data is transferred, ICF is cleared. 0 Transfer is in progress. 1 Transfer is complete, and set by the falling edge of the ninth clock of a byte transfer.
6 IAAS	I²C addressed as a slave bit. The CPU is interrupted if the interrupt enable (I2CR[IIEN]) is set. The CPU must check the slave read/write bit (SRW) and set its TX/RX mode accordingly. Writing to I2CR clears this bit. 0 Not addressed 1 Addressed as a slave. Set when its own address (IADR) matches the calling address.
5 IBB	I²C bus busy bit. Indicates the status of the bus. 0 Bus is idle. If a STOP signal is detected, IBB is cleared. 1 Bus is busy. When START is detected, IBB is set.
4 IAL	Arbitration lost. Set by hardware in the following circumstances (IAL must be cleared by software by writing a “0” to it): • SDA input sampled low when the master drives high during an address or data-transmit cycle. • SDA input sampled low when the master drives high during the acknowledge bit of a data-receive cycle. For the above two cases, the bit is set at the falling edge of 9th SCL clock during the ACK cycle. • A start cycle is attempted when the bus is busy. • A repeated start cycle is requested in slave mode. • A stop condition is detected when the master did not request it. **Note:** Software cannot set the bit. 0 No arbitration lost. 1 Arbitration is lost.

Table 40-10. I²C Status Register Field Descriptions

Field	Description
3	Reserved
2 SRW	Slave read/write. When the I²C is addressed as a slave, IAAS is set, and the slave read/write bit (SRW) indicates the value of the R/W command bit of the calling address sent from the master. SRW is valid only when a complete transfer has occurred, no other transfers have been initiated, and the I²C module is a slave and has an address match. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IIF	I²C interrupt. Must be cleared by the software by writing a “0” to it in the interrupt routine. **Note:** The software cannot set the bit. 0 No I²C interrupt pending. 1 An interrupt is pending. This causes a processor interrupt request (if the interrupt enable is asserted [IIEN = 1]). The interrupt is set when one of the following occurs: • One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock). • An address is received that matches its own specific address in slave-receive mode. • Arbitration is lost.
0 RXAK	Received acknowledge. This is the value received of the SDA input for the acknowledge bit during a bus cycle. 0 An “acknowledge” signal was received after the completion of an 8-bit data transmission on the bus. 1 A “No acknowledge” signal was detected at the ninth clock.

40.3.3.5 I²C Data Register (I2DR)

In master-receive mode, reading the data register (I2DR) allows a read to occur and initiates the next byte to be received. In slave mode, the same function is available after it is addressed. Figure 40-11 shows the I²C Data Register; Table 40-9 provides its field descriptions.

0xBASE+0x010(I2DR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DATA							
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 40-11. I²C Data Register

Table 40-9. I²C Data Register Field Descriptions

Field	Description
15–8	Reserved
7–0 DATA	Data Byte. Holds the last data byte received or the next data byte to be transferred. Software writes the next data byte to be transmitted or reads the data byte received.

NOTE

The core-written value in I2DR cannot be read back by the core: Only data written by the I²C bus side can be read.

40.4 Functional Description

This contains the following subsections:

- Section 40.4.1, I^{2}C System Configuration”
- Section 40.4.2, I^{2}C Protocol”
- Section 40.4.3, Arbitration Procedure”
- Section 40.4.4, Clock Synchronization”
- Section 40.4.5, Handshaking”
- Section 40.4.6, Clock Stretching”
- Section 40.4.7, IP Bus Accesses”
- Section 40.4.8, Generation of Transfer Error on IP Bus”

40.4.1 I^{2}C System Configuration

Out of a reset, the I^{2}C module defaults to slave receive operations. Thus, when not operating as a master or responding to a slave transmit address, the I^{2}C module defaults to the slave receiver state. For exceptions, see Section 40.5, Initialization/Application Information.”

NOTE

The I^{2}C module is designed to be compatible with the Philips I^{2}C bus protocol. For information on system configuration, protocol, and restrictions, refer to the I^{2}C Bus Specification, Version 2.1. The I^{2}C module supports Standard and Fast modes only.

40.4.2 I^{2}C Protocol

The I^{2}C communication protocol consists of six components, as follows:

- START
- Data Source/Recipient
- Data Direction
- Slave Acknowledge
- Data Acknowledge
- STOP

See Figure 40-12 for the I^{2}C standard communication protocol, as defined in the following sections.

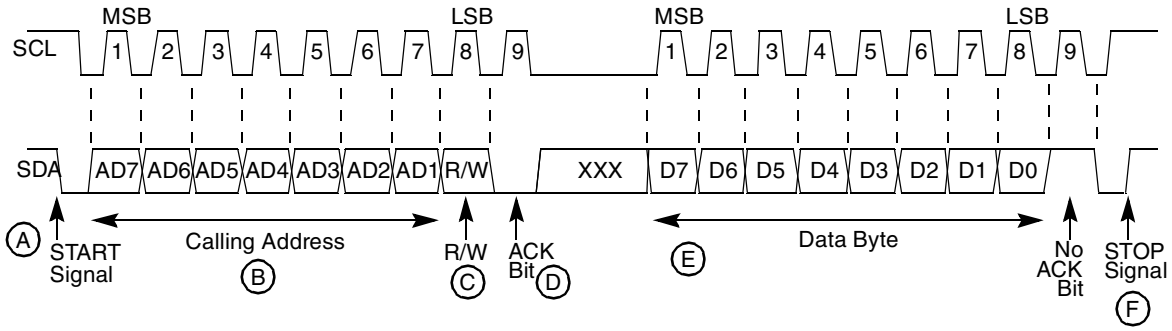


Figure 40-12. I^{2}C Standard Communication Protocol

40.4.2.1 START Signal

When no other device is a bus master (both SCL and SDA lines are at logic high), a device can initiate communication by sending a START signal (see A in Figure 40-12). A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a data transfer (each data transfer can be several bytes long) and awakens all slaves.

40.4.2.2 Slave Address Transmission

The master sends the slave address in the first byte after the START signal (B). After the seven-bit calling address, it sends the R/W bit (C), which tells the slave data transfer direction.

Each slave must have a unique address. An I^{2}C master must not transmit an address that is the same as its slave address; it cannot be master and slave at the same time.

The slave whose address matches that sent by the master pulls SDA low at the ninth clock (D) to return an acknowledge bit.

40.4.2.3 Data Transfer

When successful slave addressing is achieved, the data transfer can proceed (E) on a byte-by-byte basis in the direction specified by the R/W bit sent by the calling master.

Data can be changed only while SCL is low and must be held stable while SCL is high, as shown in Figure 40-12. SCL is pulsed once for each data bit, with the mishap being sent first. The receiving device must acknowledge each byte by pulling SDA low at the ninth clock; therefore, a data byte transfer takes nine clock pulses.

If it does not acknowledge the master, the slave receiver must leave SDA high. The master can then generate a STOP signal to abort the data transfer or generate a START signal (a repeated start, as shown in Figure 40-13) to start a new calling sequence.

If the master receiver does not acknowledge the slave transmitter after a byte transmission, it means end-of-data to the slave. The slave releases SDA for the master to generate a STOP or START signal.

40.4.2.4 STOP Signal

The master can terminate communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is at logical high (F).

NOTE

A master can generate a STOP even if the slave has made an acknowledgment; at which point, the slave must release the bus.

40.4.2.5 Repeat Start

Instead of signalling a STOP, the master can repeat the START signal, followed by a calling command (see A in Figure 40-13). A repeated START occurs when a START signal is generated without first generating a STOP signal to end the communication. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.

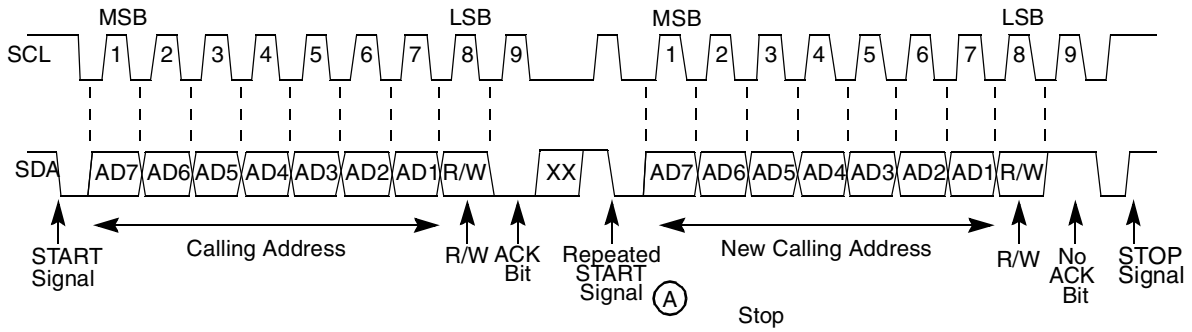


Figure 40-13. Repeated START

40.4.3 Arbitration Procedure

If multiple devices simultaneously request the bus, the bus clock is determined by a synchronization procedure in which the low period equals the longest clock-low period among the devices, and the high period equals the shortest. A data arbitration procedure determines the relative priority of competing devices. A device loses arbitration if it sends logic high while another sends logic low; it immediately switches to slave-receive mode and stops driving SDA. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets the arbitration lost bit in the I^2C Status register (I2SR[IAL] to indicate loss of arbitration.

40.4.4 Clock Synchronization

Because wire-AND logic is used, a high-to-low transition on SCL affects devices connected to the bus. Devices start counting their low period when the master drives SCL low. When a device clock goes low, it holds SCL low until the clock high state is reached. However, the low-to-high change in this device clock may not change the state of SCL if another device clock is still in its low period. Therefore, the device with the longest low period holds the synchronized clock SCL low. Devices with shorter low periods enter a high wait state during this time (see Figure 40-14). When all devices involved have counted off their low period, the synchronized clock SCL is released and pulled high. There is then no difference between device

clocks and the state of SCL, so all of the devices start counting their high periods. The first device to complete its high period pulls SCL low again.

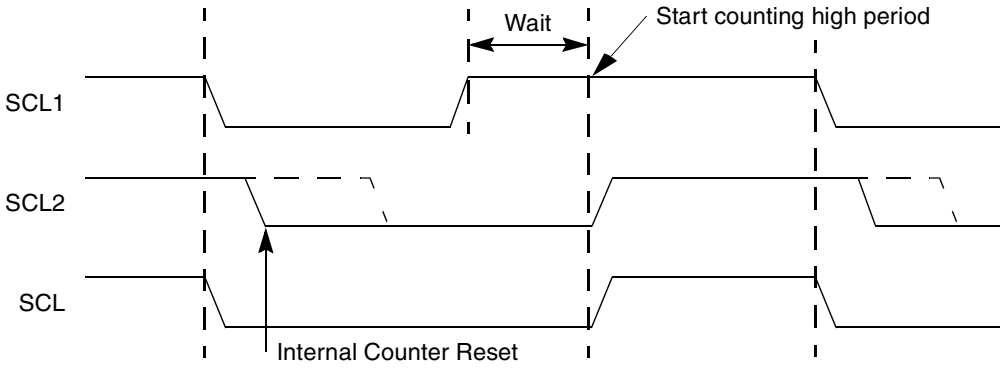


Figure 40-14. Synchronized Clock SCL

40.4.5 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. Slave devices can hold SCL low after completing one byte transfer (9 bits). In such a case, the clock mechanism halts the bus clock and forces the master clock into a wait state until the slave releases SCL.

40.4.6 Clock Stretching

Slaves can use the clock synchronization mechanism to slow down the transfer bit rate. After the master has driven SCL low, the slave can drive SCL low for the required period and then release it. If the slave SCL low period is longer than the master SCL low period, the resulting SCL bus signal low period is stretched.

40.4.7 IP Bus Accesses

I^2C is a 16-bit IP module. Only half-word accesses should be performed to the module.

40.4.8 Generation of Transfer Error on IP Bus

If an address is received on the IP slave bus interface that is not implemented, an access error is generated (ips_xfr_err is asserted). The input pin resp_sel provides the configuration capability to generate this response. The resp_sel pin must be asserted to enable the ips_xfr_err signal.

40.5 Initialization/Application Information

Before the interface can transfer serial data, registers must be initialized, as follows:

1. Set the data sampling rate (IFDR[IC] to obtain SCL frequency from the system bus clock.
2. Update the address in the (IADR) to define its slave address (address can range from 0 to 0x7f).
3. Set the I^2C enable bit (I2CR[IEN]) to enable the I^2C bus interface system.

4. Modify the bits in the I^2CR to select master/slave mode, transmit/receive mode, and interrupt-enable or not.

40.5.1 Generation of START

After completion of the initialization procedure, serial data can be transmitted by selecting the master transmitter mode. On a multiple-master bus system, the busy bus (I2SR[IBB]) must be tested to determine whether the serial bus is free. If the bus is free (IBB = 0), the START signal and the first byte (the slave address) can be sent. The data written to the data register comprises the address of the desired slave and the LSB indicates the transfer direction.

The free time between a STOP and the next START condition is built into the hardware that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period, it may be necessary to wait until the I^2C is busy after writing the calling address to the data register (I2DR) before proceeding to load data into the data register (I2DR).

40.5.2 Post-Transfer Software Response

Sending or receiving a byte sets the data transferring bit (I2SR[ICF]), which indicates one byte communication is finished. Upon completion, the interrupt status (I2SR[IIF]) is also set. An external interrupt is generated if the interrupt enable (I2CR[IIEN]) is set. The software must first clear the interrupt status (I2SR[IIF]) in the interrupt routine. (See the flow chart in Figure 40-15.) The data transferring bit (I2SR[ICF]) is cleared either by reading from I2DR in receive mode or by writing to this register in transmit mode.

The software can service the I^2C I/O in the main program by monitoring the interrupt status (I2SR[IIF]) if the interrupt enable is de-asserted. In this case, the interrupt status should be polled of the data transferring bit (I2SR[ICF]) because the operation is different when arbitration is lost.

When an interrupt occurs at the end of the address cycle, the master is always in transmit mode; that is, the address is sent. If master receive mode is required, then (I2DR[R/W], I2CR[MTX] should be toggled.

During slave-mode address cycles (I2SR[IAAS] = 1), the slave read/write bit I2SR[SRW] is read to determine the direction of the next transfer. The transmit/receive bit (I2CR[MTX]) should also be programmed accordingly. For slave-mode data cycles (IAAS = 0), SRW is invalid. MTX should be read to determine the current transfer direction.

40.5.3 Generation of STOP

A data transfer ends when the master signals a STOP, which can occur after all data is sent.

For a master receiver to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last data byte. This is done by setting the transmit acknowledge bit (I2CR[TXAK]) before reading the next-to-last byte. Before the last byte is read, a STOP signal must be generated.

40.5.4 Generation of Repeated START

After the data transfer, if the master still wants the bus, it can signal another START followed by another slave address without signalling a STOP.

40.5.5 Slave Mode

In the slave interrupt service routine (see Figure 40-15), the module addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the transmit/receive mode select bit (I2CR[MTX]) according to the I2SR[SRW]. Writing to the I2CR clears the IAAS automatically. The only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred; interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer can now be initiated by writing information to I2DR for slave transmits, or read from I2DR in slave-receive mode. A dummy read of I2DR in slave/receive mode releases SCL, allowing the master to send data.

In the slave transmitter routine, the receive acknowledge bit (I2SR[RXAK]) must be tested before sending the next byte of data. Setting RXAK means an end-of-data signal from the master receiver, after which the software must switch it from transmitter to receiver mode. Reading the data register (I2DR) then releases SCL so that the master can generate a STOP signal.

40.5.6 Arbitration Lost

If several devices try to engage the bus at the same time, one becomes master. Hardware immediately switches devices that lose arbitration to slave receive mode. Data output to SDA stops, but SCL is still generated until the end of the byte during which arbitration is lost. An interrupt occurs at the falling edge of the ninth clock of this transfer if the arbitration is lost (I2SR[IAL] = 1), and the slave mode is selected (I2CR[MSTA] = 0). See the flow chart in Figure 40-15.

If a device that is not a master tries to transmit or do a START, hardware inhibits the transmission, clears MSTA without signalling a STOP, generates an interrupt to the CPU, and sets IAL to indicate a failed attempt to engage the bus. When considering these cases, the slave service routine should first test IAL, and the software should clear it if it is set.

For Multi-master mode, when an I^2C module is enabled when the bus is busy and asserts START, the IAL bit gets set only for SDA=0, SCL=0/1, SDA=1, and SCL=0, but not for SDA=1 and SCA=1, which is the same as bus idle state.

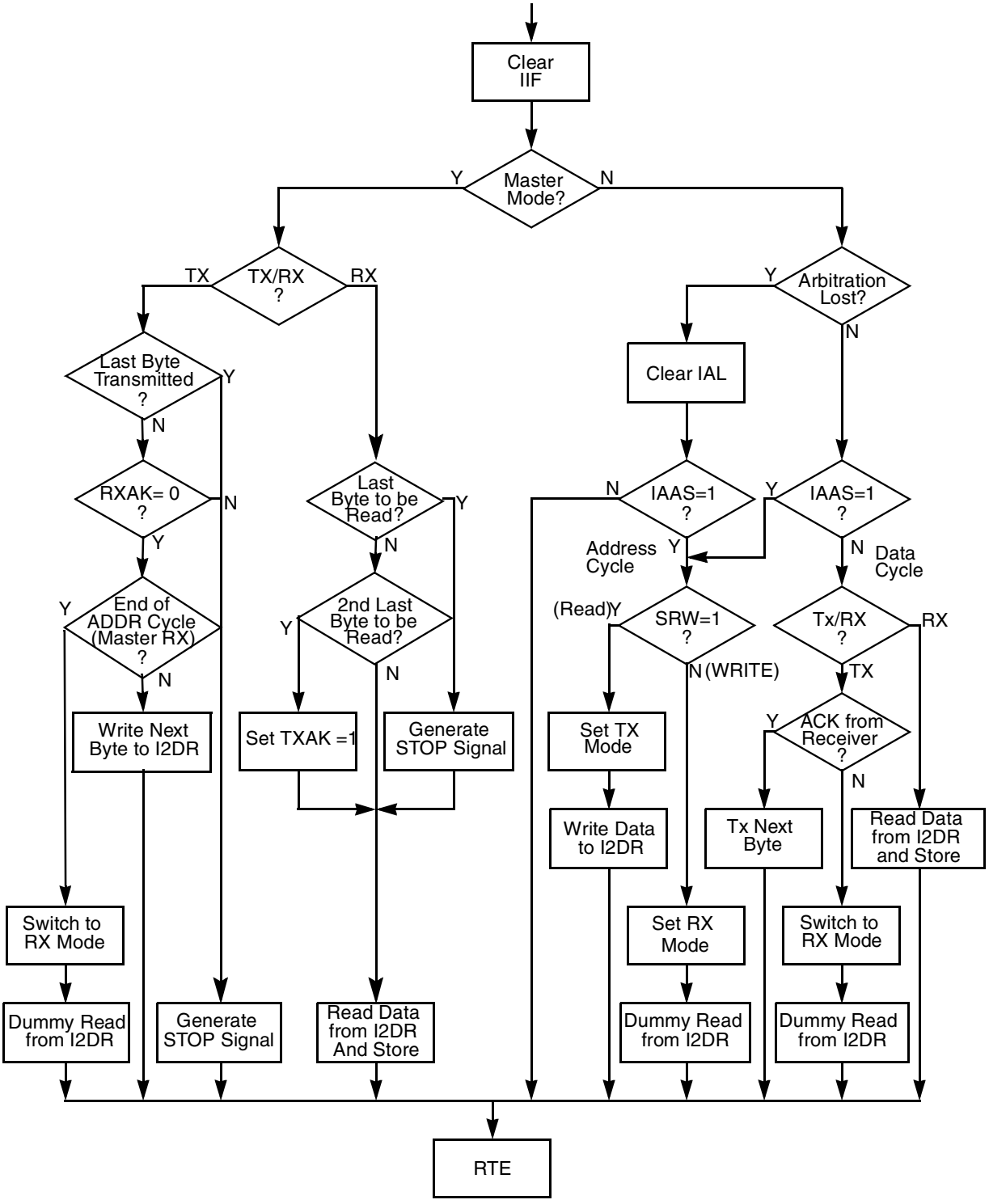


Figure 40-15. Flow-Chart of Typical I^2C Interrupt Routine

NOTE

For a repeated start-only, the stop generation stage will not occur in master mode. A loop will repeat itself without stopping for the next start.

40.5.7 Timing Section

Please refer to I^2C module electrical specification document for the I^2C module timing characteristics.

40.5.8 Software Restrictions

Software should take care that there is a delay of at least 2 ipg_clk_patref cycles after it sets the RSTA bit of I^2C control register before writing to the I^2C data register. The maximum possible clock period of ipg_clk_patref is 78 ns.

Chapter 41
IC Identification (IIM)

The IIM provides an interface for reading and in some cases programming and/or overriding identification and control information stored in on-chip fuse elements. The module supports electrically-programmable poly fuses (e-Fuses).

The IIM also provides a set of volatile software-accessible signals which can be used for software control of hardware elements, not requiring non-volatility.

41.1 Overview

The IIM provides the primary user-visible mechanism for interfacing with on-chip fuse elements. Among the uses for the fuses are unique chip identifiers, mask revision numbers, cryptographic keys, and various control signals requiring permanent non-volatility. The IIM also provides up to 28 volatile control signals

41.1.1 Modes of Operation

The IIM is in its functional mode (all specified functionality available) any time it is out of reset and supplied with the proper clocks.

41.2 Memory Map and Register Definition

Section 41.2.3, Register Descriptions," provides the detailed register descriptions for all of the IIM registers

41.2.1 Memory Map

All registers are 8-bit wide, but addressable on 32-bit boundaries. Only the bottom 8 bits (the usable bits) of each register are shown in the following diagrams. The top 24 bits always read as 0 and writes to them are ignored. Table 41-7 shows the IIM memory map. For a description of all of the addressable fuses, please refer to Chapter 6, "Fuse Map in Book I.

All IIM registers can be accessed in the user mode, except FCTL, which controls programming functions and can be accessed in supervisor mode only.

Table 41-2. IIM Memory Map

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x000 (STAT)	Status register	R/W	— —	41.2.3.1/41-4
0xBASE+0x004 (STATM)	Status IRQ Mask register	R/W	00	41.2.3.2/41-5

Table 41-2. IIM Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x008 (ERR)	Module Errors register	R/W	0 —	41.2.3.3/41-5
0xBASE+0x00C (EMASK)	Error IRQ Mask register	R/W	0 —	41.2.3.4/41-7
0xBASE+0x010 (FCTL)	Fuse Control register	R/W	30	41.2.3.5/41-8
0xBASE+0x014 (UA)	Upper Address register	R/W	— 0	41.2.3.6/41-9
0xBASE+0x018 (LA)	Lower Address register	R/W	00	41.2.3.7/41-10
0xBASE+0x01C (SDAT)	Explicit Sense Data register	read-only	00	41.2.3.8/41-10
0xBASE+0x020 (PREV)	Product Revision register	read-only	— —	41.2.3.9/41-11
0xBASE+0x024 (SREV)	Silicon Revision register	read-only	— —	41.2.3.10/41-11
0xBASE+0x028 (PREG_P)	Program Protection register	R/W	— —	41.2.3.11/41-12
0xBASE+0x02C (SCS0)	Software-Controllable Signals register 0	R/W	00	41.2.3.12/41-12
0xBASE+0x030 (SCS1)	Software_Controllable Volatile Hardware - Visible Signals register (1–3)	R/W	00	41.2.3.13/41-13
0xBASE+0x034 (SCS2)		R/W	00	
0xBASE+0x030 (SCS1)		R/W	00	
0xBASE+0x03C (SaharaEn0)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.15/41-15
0xBASE+0x040 (SaharaEn1)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.16/41-16
0xBASE+0x044 (SaharaEn2)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.17/41-16
0xBASE+0x048 (SaharaEn3)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.18/41-17
0xBASE+0x04C (SaharaEn4)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.19/41-17
0xBASE+0x050 (SaharaEn5)	Software setting. Upon setting proper value allows SAHARA_EN blowing.	R/W	00	41.2.3.20/41-18

41.2.2 Register Summary

Figure 41-1 shows the key to the register fields and Table 41-2 shows the register figure conventions.

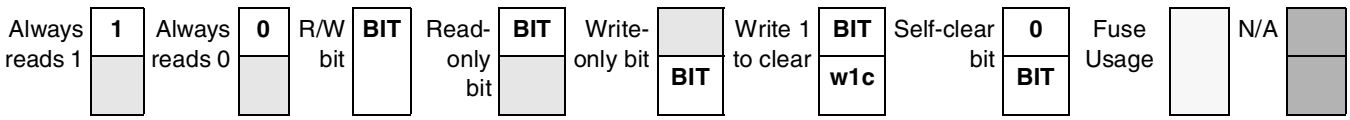


Figure 41-1. Key to Register Fields

Table 41-3 shows the IIM register summary. For detailed information about the fuses in the i.MX51, see Chapter 6, “Fuse Map,”

Table 41-3. IIM Register Summary

Bank	Name		7	6	5	4	3	2	1	0
	0xBASE+0x000 (STAT)	R	BUSY						PRGD	SNSD
		W							w1c	w1c
	0xBASE+0x004 (STATM)	R							PRGD_M	SNSD_M
		W								
	0xBASE+0x008 (ERR)	R	PRGE	WPE	OPE	RPE	WLRE	SNSE	PARITYE	
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	
	0xBASE+0x00C (EMASK)	R	PRGE_M	WPE_M	OPE_M	RPE_M	WLRE_M	SNSE_M	PARITY_M	
		W								
	0xBASE+0x010 (FCTL)	R	DPC	PRG_LENGTH			ESNS_N	ESNS_0	ESNS_1	PRG
		W								
	0xBASE+0x014 (UA)	R			Address[13:8]					
		W								
	0xBASE+0x018 (LA)	R	Address[7:0]							
		W								
	0xBASE+0x01C (SDAT)	R	Data[7:0]							
		W								
	0xBASE+0x020 (PREV)	R	PRODUC_ REV[4:0]					PRODUCT_VT[2:0]		
		W								
	0xBASE+0x024 (SREV)	R	SILICON_REV[7:0]							
		W								
	0xBASE+0x028 (PREG_P)	R	PROTECTION_REG[7:0]							
		W								
	0xBASE+0x02C (SCS0)	R		HAB_JDE	SCS[26:21]					
		W	LOCK							
	0xBASE+0x030 (SCS1)	R	LOCK	SCS[20:14]						
		W								
	0xBASE+0x034 (SCS2)	R	LOCK	SCS[13:7]						
		W								
	0xBASE+0x030 (SCS1)	R	LOCK	SCS[6:0]						

41.2.3 Register Descriptions

This section contains the detailed register descriptions for the IIM registers.

41.2.3.1 Status Register (STAT)

All module status information is read using the STAT register.

See Figure 41-2 for illustration of valid bits in the Status Register and Table 41-4 for description of the bit fields.

0xBASE+0x000 (STAT) Access: Supervisor read-only

	7	6	5	4	3	2	1	0
R	BUSY						PRGD	SNSD
W							w1c	w1c
RESET	0	0	—	—	—	—	0	0

Figure 41-2. Status Register

Table 41-4. Status Register Field Descriptions

Field	Description
7 BUSY	Indicates whether the IIM is busy with a program or sense cycle. Any attempt to access the IIM registers other than STAT while it is busy with a program or sense cycle (BUSY asserted) results in a bus error. 0 The IIM is not busy with a program or sense cycle 1 The IIM is busy with a program or sense cycle
6–2	N/A
1 PRGD	Program Done. Indicates an e-Fuse program operation is done. Assertion causes an interrupt request (irq_b signal asserted) if PRGD_M is set in the Status IRQ Mask Register. This bit is automatically set by hardware upon completion of an e-Fuse program cycle; software must clear the bit by writing 1 to it. 0 Program operation has not finished (read); no meaning (write) 1 Program operation has finished (read); clear bit (write)
0 SNSD	Explicit Sense Cycle Done. Indicates that an explicit fuse sense cycle is done, and the data is available in SDAT. Assertion causes an interrupt request if SNSD_M is set in the Status IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 No explicit sense cycle has finished (read); no meaning (write) 1 An explicit sense cycle has finished (read); clear bit (write)

41.2.3.2 Status IRQ Mask (STATM)

See Figure 41-3 for illustration of valid bits in the Status IRQ Mask Register and Table 41-5 for description of the bit fields.

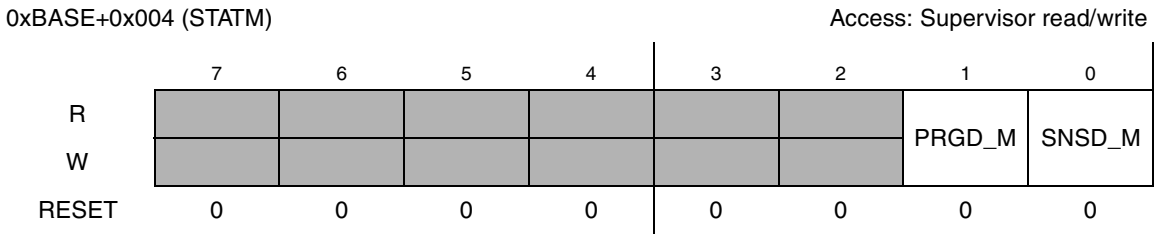


Figure 41-3. Status IRQ Mask Register

Table 41-5. Status IRQ Mask Register Field Descriptions

Field	Description
7–2	N/A
1 PRGD_M	Program Mask. Masks or unmasks IRQ generation due to PRGD events. 0 PRGD events do not cause an IRQ 1 PRGD events cause an IRQ
0 SNSD_M	Explicitly Sense Cycle Done Mask. Masks or unmasks IRQ generation due to SNSD events. 0 SNSD events do not cause an IRQ 1 SNSD events cause an IRQ

41.2.3.3 Module Errors Register (ERR)

See Figure 41-4 for illustration of valid bits in the Module Errors Register and Table 41-6 for description of the bit fields.

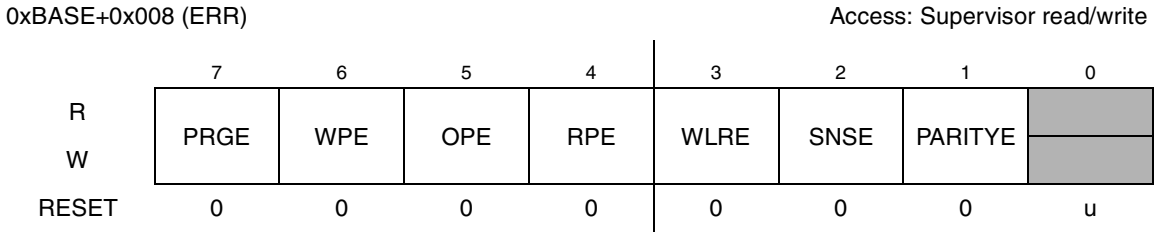


Figure 41-4. Module Errors Register

Table 41-6. Module Errors Register Field Descriptions

Field	Description
7 PRGE	Program Error. Indicates an e-Fuse program operation ended in failure. Assertion causes an interrupt request if PRGE_M is set in the Errors IRQ Mask Register. A program failure occurs when an attempt is made to program laser fuses. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 Program operation has not finished with error (read); no meaning (write) 1 Program operation has finished with an error (read); clear bit (write)
6 WPE	Write Protect Error. Indicates an e-Fuse program operation was attempted to a write-protected fuse bank, or a locked words, or when the value of PRG_P is not 8'hAA. Assertion causes an interrupt request if WPE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no write-protect error (read); no meaning (write) 1 There was a write-protect error (read); clear bit (write)
5 OPE	Override Protect Error. Indicates an attempt was made to override the values in an override-protected fuse bank, or a locked words. Assertion causes an interrupt request if OPE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no override-protect error (read); no meaning (write) 1 There was an override-protect error (read); clear bit (write)
4 RPE	Read Protect Error. Indicates an attempt was made to read values from a read-protected fuse bank or SCC. Assertion causes an interrupt request if RPE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no read-protect error (read); no meaning (write) 1 There was a read-protect error (read); clear bit (write)
3 WLRE	Write to Locked Register Error. Indicates an attempt was made to write to a locked SCS register. Assertion causes an interrupt request if WLRE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no write-to-locked-register error (read); no meaning (write) 1 There was a write-to-locked-register error (read); clear bit (write)
2 SNSE	Explicit Sense Cycle Error. Indicates that an explicit fuse sense was refused, because FBESP is set to 1, or more than two bits of SNS_N, SNS_1, SNS_0, PRG are asserted at the same moment. Assertion causes an interrupt request if SNSE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no explicit sense error (read); no meaning (write) 1 There was an explicit sense error (read); clear bit (write)
1 PARITYE	Parity Error of Cache. Indicate that an parity error was detected in hardware fuse cache or software fuse cache. Assertion causes an interrupt request if PARITYE_M is set in the Errors IRQ Mask Register. This bit is automatically set by hardware and must be cleared by software by writing 1 to it. 0 There was no explicit sense error (read); no meaning (write) 1 There was an explicit sense error (read); clear bit (write)
0	N/A

41.2.3.4 Error IRQ Mask Register (EMASK)

See Figure 41-5 for illustration of valid bits in the Error IRQ Mask Register and Table 41-7 for description of the bit fields.

0xBASE+0x00C (EMASK) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R	PRGE_M	WPE_M	OPE_M	RPE_M	WLRE_M	SNSE_M	PARITYE_M	
W								
RESET	0	0	0	0	0	0	0	u

Figure 41-5. Error IRQ Mask Register

Table 41-7. Error IRQ Mask Register Field Descriptions

Field	Description
7 PRGE_M	Program Error Mask. Masks or unmasks IRQ generation due to PRGE events. 0 PRGE events do not cause an IRQ 1 PRGE events cause an IRQ
6 WPE_M	Write Protect Error Mask. Masks or unmasks IRQ generation due to WPE events. 0 WPE events do not cause an IRQ 1 WPE events cause an IRQ
5 OPE_M	Override Protect Error Mask. Masks or unmasks IRQ generation due to OPE events. 0 OPE events do not cause an IRQ 1 OPE events cause an IRQ
4 RPE_M	Read Protect Error Mask. Masks or unmasks IRQ generation due to RPE events. 0 RPE events do not cause an IRQ 1 RPE events cause an IRQ
3 WLRE_M	Write to Locked Register Error Mask. Masks or unmasks IRQ generation due to WLRE events. 0 WLRE events do not cause an IRQ 1 WLRE events cause an IRQ
2 SNSE_M	Explicit Sense Cycle Error Mask. Masks or unmasks IRQ generation due to SNSE events. 0 SNSE events do not cause an IRQ 1 SNSE events cause an IRQ
1 PARITYE_M	Parity Error of Cache Mask. Masks or unmasks IRQ generation due to PARITYE events. 0 PARITYE events do not cause an IRQ 1 PARITYE events cause an IRQ
0	N/A

41.2.3.5 Fuse Control Register (FCTL)

See Figure 41-6 for illustration of valid bits in the Fuse Control Register and Table 41-8 for description of the bit fields.

0xBASE+0x010 (FCTL) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R W	DPC	PRG_LENGTH[2:0]			ESNS_N	ESNS_0	ESNS_1	PRG
RESET	0	0	1	1	0	0	0	0

Figure 41-6. Fuse Control Register

Table 41-8. Fuse Control Register Field Descriptions

Field	Description
7 DPC	Delayed Program Cycle. This bit is a control bit, selecting immediate or delayed program. When this bit is cleared, program cycles start immediately upon setting of PRG bit. When this bit is asserted, program cycles is delayed until input signal delayed_pgm_start is asserted. 0 Program cycles begin immediately upon setting of the PRG bit 1 Program cycles are delayed; they do not begin until delayed_pgm_start signal is asserted
6–4 PRG_LENGTH[2:0]	Program Length. This 3 bits define the length of program pulse PRG_LENGTH*(period of 32k clock) Setting is unknown.
3 ESNS_N	Explicit Sense - Normal. Writing 1 to this bit initiate an unstressed (normal) explicit sense cycle. Read of this bit always returns zero. FSM generate a “done” signal when the operation complete. This bit is cleared automatically by hardware when sense operation completed. Only one of ESNS_N, ESNS_0, ESNS_1, PRG can be asserted, otherwise ESNS_E will be asserted to indicate this error. 0 Return 0 for all read (read); No meaning (write) 1 Initiate an unstressed explicit sense cycle (write)
2 ESNS_0	Explicit Sense - 0-stressed. Writing 1 to this bit initiate a 0-stressed explicit sense cycle. Read of this bit always returns zero.FSM generate a “done” signal when the operation complete. This bit is cleared automatically by hardware when sense operation completed. During 0-stressed explicit sense cycles, the **epm_readsense0** signal is asserted to the fuse banks. Only one of ESNS_N, ESNS_0, ESNS_1, PRG can be asserted, otherwise ESNS_E will be asserted to indicate this error. 0 Return 0 for all read (read); No meaning (write) 1 Initiate a 0-stressed explicit sense cycle (write)

Table 41-8. Fuse Control Register Field Descriptions (continued)

Field	Description
1 ESNS_1	Explicit Sense - 1-stressed. Writing 1 to this bit initiate a 1-stressed explicit sense cycle. Read of this bit always returns zero. FSM generate a “done” signal when the operation complete. This bit is cleared automatically by hardware when sense operation completed. During 1-stressed explicit sense cycles, the **epm_readsense1** signal is asserted to the fuse banks. Only one of ESNS_N, ESNS_0, ESNS_1, PRG can be asserted, otherwise ESNS_E will be asserted to indicate this error. Only one of ESNS_N, ESNS_0, ESNS_1, PRG can be asserted, otherwise ESNS_E will be asserted to indicate this error. 0 Return 0 for all read (read); No meaning (write) 1 Initiate a 1-stressed explicit sense cycle (write)
0 PRG	Program. Writing 1 to this bit initiate a fuse program cycle. Read of this bit always returns zero. FSM generate a “done” signal when the operation complete. This bit is cleared automatically by hardware when program operation completed. Only one of ESNS_N, ESNS_0, ESNS_1, PRG can be asserted, otherwise ESNS_E will be asserted to indicate this error. 0 Return 0 for all read (read); No meaning (write) 1 Initiate a program cycle (write)

41.2.3.6 Upper Address (UA)

The top part of the address of the e-Fuse bit to be programmed or the word to be sensed in an explicit sense cycle. Note that programming is done by bit, so the program address is a full-bit address. Sensing is done on 8-bit words, so the bottom three bits of the address are ignored.

See Figure 41-7 for illustration of valid bits in the Upper Address Register and Table 41-9 for description of the bit fields.

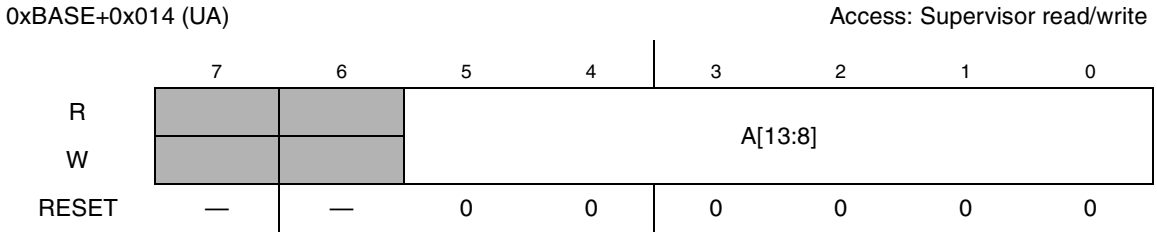


Figure 41-7. Upper Address Register

Table 41-9. Upper Address Register Field Descriptions

Field	Description
7–6	N/A
5–0 A[13–8]	The top six bits of the address of the e-Fuse bit to be programmed or the word to be sensed explicitly. The address must be written prior to setting the PRG or ESNS_x bit in FCTL to initiate the program/sense operation. A[13–11] select the Fuse bank. A[10–8] provide the most significant portion of the row address within the bank.

41.2.3.7 Lower Address (LA)

The bottom 8 bits of the address of the e-Fuse bit to be programmed or word to be explicitly sensed.

See Figure 41-8 for illustration of valid bits in the Lower Address Register and Table 41-10 for description of the bit fields.

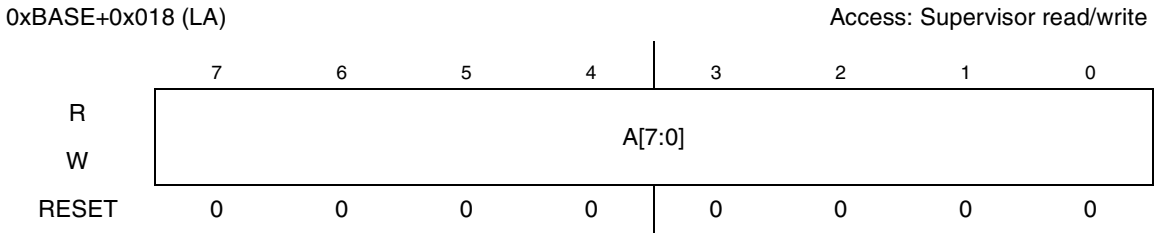


Figure 41-8. Lower Address Register

Table 41-10. Lower Address Register Field Descriptions

Field	Description
7–0 A	The bottom eight bits of the address of the e-Fuse bit to be programmed or word to be sensed explicitly. The address must be written prior to setting the PRG or ESNS_x bit in FCTL to initiate a program or sense operation. A[7:3] provides the least significant portion of the row address. A[2:0] select the bit position within the selected row.

41.2.3.8 Explicit Sense Data (SDAT)

See Figure 41-9 for illustration of valid bits in the Explicit Sense Data Register and Table 41-11 for description of the bit fields.

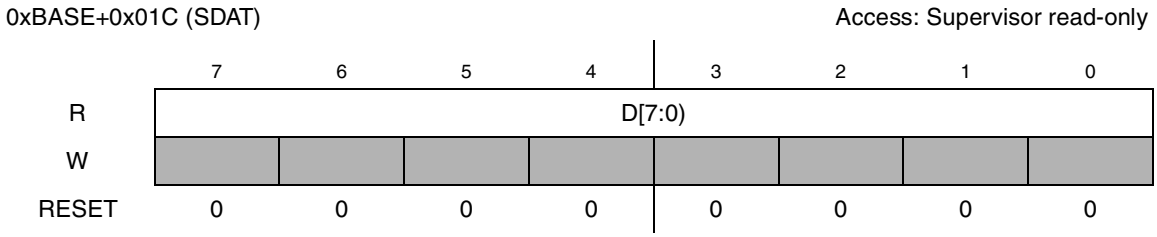


Figure 41-9. Explicit Sense Data Register

On an explicit sense cycle, the data sensed from the fuses is placed in this register at the conclusion of the sense cycle. Software can recognize the conclusion of the sense cycle by the assertion of SNSD in STAT.

Table 41-11. Explicit Sense Data Register Field Descriptions

Field	Description
7–0 D	The data sensed from the fuses. Setting is unknown.

41.2.3.9 Product Revision (PREV)

The product revision. This corresponds to the top eight bits of the deprecated HW_REV register.

See Figure 41-10 for illustration of valid bits in the Product Revision Register and Table 41-12 for description of the bit fields.

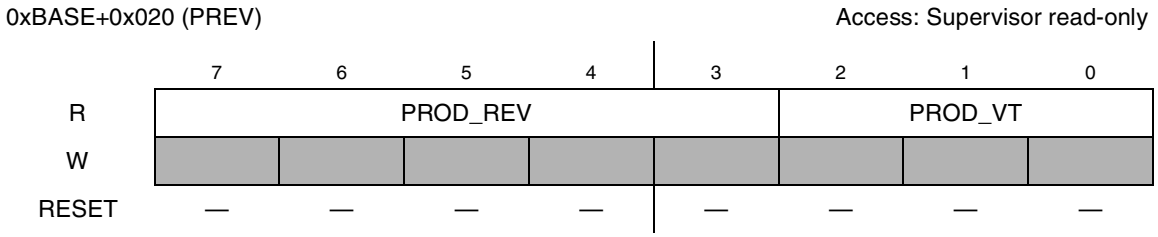


Figure 41-10. Product Revision Register

Table 41-12. Product Revision Register Field Descriptions

Field	Description
7–3 PROD_REV	Product Revision. The product revision (specific to the product or product family). Setting according to product revision.
2–0 PROD_VT	Product vendor or technology. The product vendor and/or technology (specific to the product or product family). Setting according to product vendor/technology.

41.2.3.10 Silicon Revision (SREV)

The silicon revision (that is, mask revision). This corresponds to the bottom 8 bits of the deprecated HW_REV register.

See Figure 41-11 for illustration of valid bits in the Silicon Revision Register and Table 41-13 for description of the bit fields.

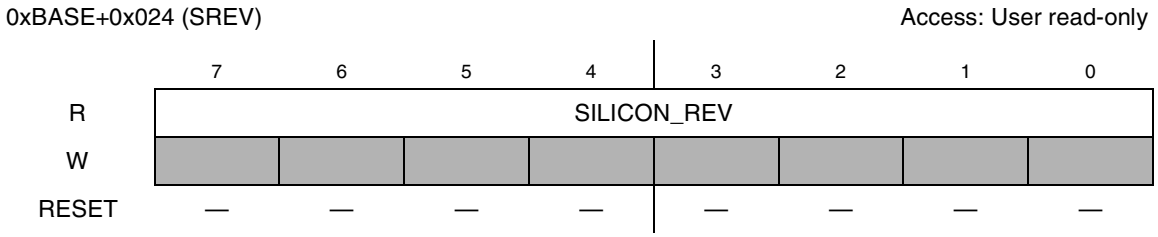


Figure 41-11. Silicon Revision Register

Table 41-13. Silicon Revision Register Field Descriptions

Field	Description
7–0 SILICON_REV	Mask Set Revision. The mask set revision used in fabrication of the part. The value changes with each change to the mask set. Setting according to mask set revision.

41.2.3.11 Program Protection Register (PRG_P)

This register is used to against accidental fuse programming. The fuses can be blown only when the value of this register is 8'hAA. Software should only program this register to 8'hAA while actively blowing fuses. After the program operation is complete, this register should be immediately reprogrammed to a different value.

See Figure 41-12 for illustration of valid bits in the Program Protection Register and Table 41-14 for description of the bit fields.

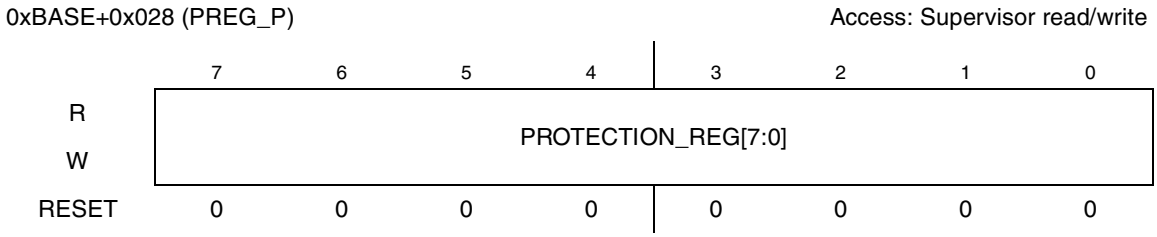


Figure 41-12. Program Protection Register

Table 41-14. Program Protection Register Field Descriptions

Field	Description
7–0 PROTECTION_REG	The fuses can be blown only when the value of this register is 8'hAA. Any attempt to program the fuse while the value is other than 8'hAA will be terminated with error. WPE bit will be asserted. Setting is undefined.

41.2.3.12 Software-Controllable Signals Register 0 (SCS0)

This register is physically located in the hardware-visible signals sub-module. It implements software-controlled, volatile signals which can drive SoC-level nets for feature enabling.

See Figure 41-13 for illustration of valid bits in the Software-Controllable Signals Register 0 and Table 41-15 for description of the bit fields.

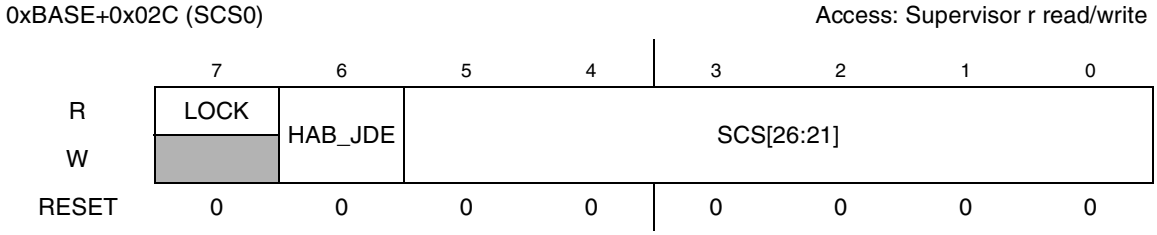


Figure 41-13. Software Controllable Signals Register 0

Table 41-15. Software Controllable Signals Register 0 Field Descriptions

Field	Description
7 LOCK	Lock this register. This bit is used to lock the contents of this register until the next reset. The intended usage is to have trusted software program the register as desired and lock it before allowing distrusted software to run. This bit is write only, read of this bit return 0. 0 The register is not locked, it may be modified 1 The register is locked, all attempts to modify it are ignored
6 HAB_JDE	HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properly signed command to do so is found and validated by the HAB. The HAB musts lock the register before passing control to the OS whether or not JTAG debugging has been enabled. Once JTAG is enabled by HAB_JDE bit, it can not be disabled unless the system is reset by POR. 0 JTAG debugging is not enabled by the HAB (it may still be enabled by other mechanisms) 1 JTAG debugging is enabled by the HAB (though this signal may be gated off)
5–0 SCS[26:21]	These bits may be used in an implementation-defined way.

41.2.3.13 Software-Controllable Signals Registers 1

This register has been dedicated for ROM boot interface with an upper SW. It is physically located in the hardware-visible signals sub-module.

0xBASE+0x030 (SCS1) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R W	LOCK	SCS[x:y]						LPB_ENABLED_STATUS
RESET	0	0	0	0	0	0	0	0

Figure 41-14. Software Controllable Signals Register 1

Table 41-16. Software Controllable Signals Register 1

Field	Description
7 LOCK	Lock this register. 0 The register is not locked, it may be modified 1 The register is locked, all attempts to modify it are ignored
6–1 SCS[x:y]	These bits may be used in an implementation-defined way
0 LPB_ENABLED_STATUS	0 – no LPB 1 – LPB

41.2.3.14 Software-Controllable Signals Registers 2–3 (SCS2–SCS3)

These registers are physically located in the hardware-visible signals submodule. They implement software-controlled, volatile signals that can drive SoC-level nets for feature enabling.

Each IIM fuse bank include a read protection bit to inhibit software fuses read and sense and to support secure computing environments. When a fuse bank's read protection bit is set (i.e.'1'), the bank's fuses cannot be read or sensed by software. The read protection bit cannot be changed if the sticky lock bit was asserted.

41.2.3.14.1 Read Lock Register on SCS2

This is the fuse bank read lock register. This corresponds to four iMX51 fuse banks.

See Figure 41-5 for illustration of valid bits in the Read Lock Register and Figure 41-17 for description of the bit fields.

0xBASE+0x034 (SCS2) Access: User read/write

	7	6	5	4	3	2	1	0
R	LOCK				FBRL3	FBRL2	FBRL1	FBRL0
W								
RESET	0	0	0	0	0	0	0	0

Figure 41-15. Read Lock Register

Table 41-17. Read Lock Register

Field	Description
7 LOCK	Lock this register. 0 The register is not locked, it may be modified 1 The register is locked, all attempts to modify it are ignored
6–4	Reserved
3 FBRL3	Read lock for fuse bank #3
2 FBRL2	Read lock for fuse bank #2
1 FBRL1	Read lock for fuse bank #1
0 FBRL0	Read lock for fuse bank #0

41.2.3.14.2 Write Lock Register on SCS3

This is the fuse bank write lock register. This corresponds to the four iMX51 fuse banks.

0xBASE+0x038 (SCS3) Access: User read/write

	7	6	5	4	3	2	1	0
R	LOCK				FBWL3	FBWL2	FBWL1	FBWL0
W								
RESET	0	0	0	0	0	0	0	0

Figure 41-16. Write Lock Register

Table 41-18. Write Lock Register

Field	Description
7 LOCK	Lock this register. 0 The register is not locked, it may be modified 1 The register is locked, all attempts to modify it are ignored
6–4	Reserved
3 FBWL3	Write lock for fuse bank #3
2 FBWL2	Write lock for fuse bank #2
1 FBWL1	Write lock for fuse bank #1
0 FBWL0	Write lock for fuse bank #0

41.2.3.15 Sahara Enable 0 (SaharaEn0)

This is the first register out of the six registers that serve as a neccesery condition for blowing up SAHRA_EN fuse.

See Figure 41-17 for illustration of valid bits in the Sahara Enable Register and Table 41-19 for description of the bit fields.

0xBASE+0x03C (SaharaEn0) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R	SaharaEn[7:0]						Suggested	
W								
RESET	0	0	0	0	0	0	0	0

Figure 41-17. Sahara Enable 0 Register

Table 41-19. Sahara Enable 0 Register Field Descriptions

Field	Description
7–0 SaharaEn[7–0]	first register out of 6 registers. assert proper value is neccesery condition for blow up SAHRA_EN fuse.

41.2.3.16 Sahara Enable 1 (SaharaEn1)

This is the second register out of the six registers that serve as a necessary condition for blowing up SAHRA_EN fuse.

See Figure 41-18 for illustration of valid bits in the Sahara Enable 1 Register and Table 41-20 for description of the bit fields.

0xBASE+0x040 (SaharaEn1) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R W	SaharaEn[15:8]							
RESET	0	0	0	0	0	0	0	0

Figure 41-18. Sahara Enable 1 Register

Table 41-20. Sahara Enable 1 Register Field Descriptions

Field	Description
7–0 SaharaEn[15–8]	second register out of 6 registers. assert proper value is necessary condition for blow up SAHRA_EN fuse.

41.2.3.17 Sahara Enable 2 (SaharaEn2)

This is the third register out of the six registers that serve as a necessary condition for blowing up SAHRA_EN fuse.

See Figure 41-19 for illustration of valid bits in the Sahara Enable 2 Register and Table 41-21 for description of the bit fields.

0xBASE+0x044 (SaharaEn2) Access: Supervisor read/write

	7	6	5	4	3	2	1	0
R W	SaharaEn[23:16]							
RESET	0	0	0	0	0	0	0	0

Figure 41-19. Sahara Enable 2 Register

Table 41-21. Sahara Enable 2 Register Field Descriptions

Field	Description
7–0 SaharaEn[23–16]	third register out of 6 registers. assert proper value is necessary condition for blow up SAHRA_EN fuse.

41.2.3.18 Sahara Enable 3 (SaharaEn3)

This is the fourth register out of the six registers that serve as a necessary condition for blowing up SAHRA_EN fuse.

See Figure 41-20 for illustration of valid bits in the Sahara Enable 3 Register and Table 41-22 for description of the bit fields.

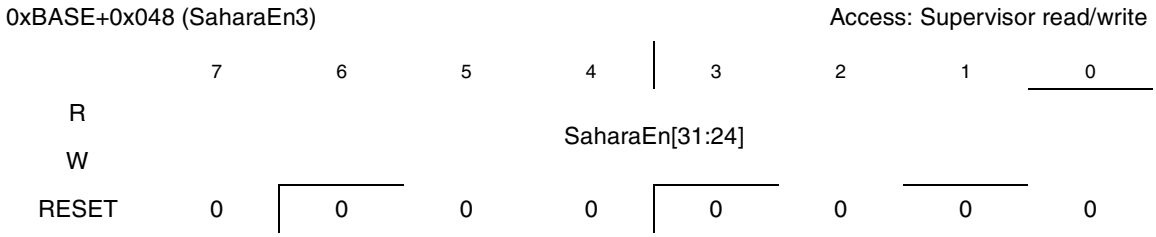


Figure 41-20. Sahara Enable 3 Register

Table 41-22. Sahara Enable 3 Register Field Descriptions

Field	Description
7–0 SaharaEn[31-24]	fourth register out of 6 registers. assert proper value is necessary condition for blow up SAHRA_EN fuse.

41.2.3.19 Sahara Enable 4 (SaharaEn4)

This is the fifth register out of the six registers that serve as a necessary condition for blowing up SAHRA_EN fuse.

See Figure 41-21 for illustration of valid bits in the Sahara Enable 4 Register and Table 41-23 for description of the bit fields.

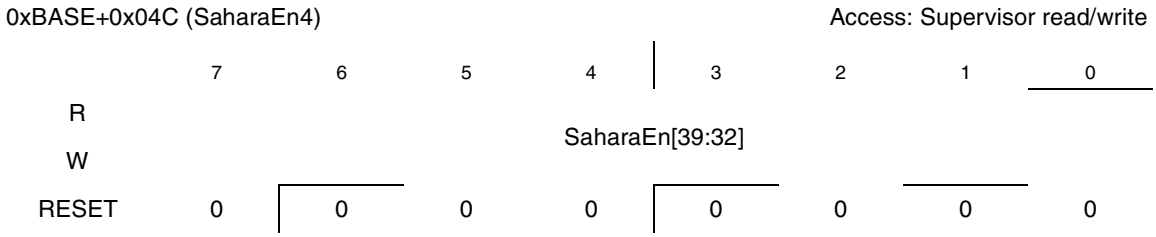


Figure 41-21. Sahara Enable 4 Register

Table 41-23. Sahara Enable 4 Register Field Descriptions

Field	Description
7–0 SaharaEn[39-32]	fifth register out of 6 registers. assert proper value is necessary condition for blow up SAHRA_EN fuse.

41.2.3.20 Sahara Enable 5 (SaharaEn5)

This is the sixth register out of the six registers that serve as a necessary condition for blowing up SAHRA_EN fuse.

See Figure 41-22 for illustration of valid bits in the Sahara Enable 5 Register and Table 41-24 for description of the bit fields.

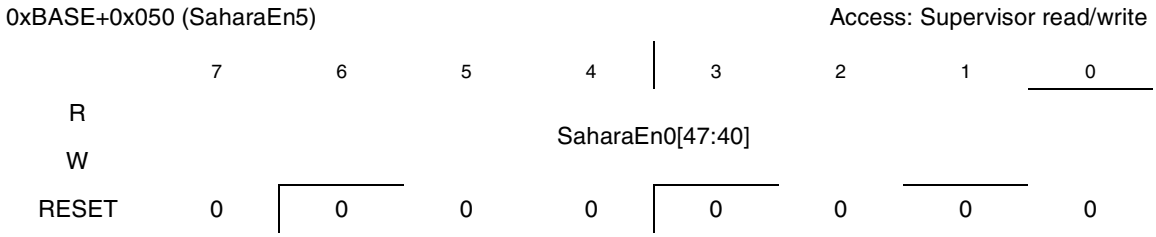


Figure 41-22. Sahara Enable 5 Register

Table 41-24. Sahara Enable 5 Register Field Descriptions

Field	Description
7–0 SaharaEn[47–40]	sixth register out of 6 registers. assert proper value is necessary condition for blow up SAHRA_EN fuse.

41.3 Functional Description

The IIM is an 8-bit IP Bus peripheral which implements interfaces for laser fuses and/or e-Fuses, as well as the hardware revision codes, cryptographic keys, and miscellaneous system-level feature enabling circuitry. Up to eight banks, each with up to 2048 fuses, are supported. Laser fuse banks and e-Fuse banks may be mixed.

41.3.1 Signal Groups

Listed below are the intended uses for the four fuse banks. Each fuse bank may contain up to 2048 fuses, and the unspecified fuses in each bank below may be used for implementation-specific purposes. However, access protection is specified on a per-bank basis; this may limit the use of the unspecified fuses.

The IIM implements non-volatile “Hardware-Visible” control signals. These are all capable of driving SoC-level nets to allow them to permanently enable or disable features in the system.

41.3.1.1 Secure JTAG Control

Among the hardware control signals implemented in Fuse Bank 0 are the Secure JTAG control bits. These fuses are used to allow or disallow JTAG access to secured resources.

Three JTAG security levels are envisioned, as shown in Table 41-25.

Table 41-25. JTAG Security Level Control Bits

Security Mode	JTAG_SMODE	JTAG_SCC_DIS	Description
No Debug	2'b11	x	The highest security level.
Secure JTAG	2'b01	x	Limit the JTAG access by using key based authentication mechanism.
JTAG Enable	2'b00	1'b1	Low Security, all JTAG features are enabled.
SCC JTAG	2'b00	1'b0	No security.

JTAG debug access may be semipermanently enabled by blowing the jtag_bp bit (JTAG Bypass Security). This, however, can be overridden by blowing the jtag_re bit (JTAG Security Re-enable). Blowing the jtag_bp bit effectively changes the security level from 'Secure JTAG' to 'JTAG Enable', without actually modifying the JTAG_SMODE fuses.

NOTE

The IIM must only allow the jtag_bp fuse to be blown if the JTAG debug security level is "Secure JTAG" (JTAG_SMODE = 2'b01) and ipt_response_in = 1, ipt_secur_block = 0. This indicates that the SJC has processed a valid response in its challenge/response protocol, and the JTAG security bypass functionality should be allowed. JTAG security is re-instated to the "Secure JTAG" level by blowing the jtag_re bit.

In addition to the three JTAG security levels, the JTAG debug notification normally passed to the Security Controller (SCC), which causes it to transition out of the secure state when debugging has become active, can be gated off using the jtag_scc fuse in JAC. If jtag_scc is left unblown, the debug active signals are masked, and do not propagate to the SCC. If jtag_scc is blown, the debug active signals propagate to the SCC as normal.

NOTE

Leaving the jtag_scc fuse unblown represents a very significant security risk, and should only be done in as few parts as possible to debug secure operation of the SCC. Once SCC debugging is complete, the jtag_scc fuse should immediately be blown. The reason for including this functionality in a fuse is to avoid the need to modify parts using FIB for SCC debugging, but the convenience afforded by the fuse must not be allowed to foster an unguarded attitude toward the dangers such insecure parts would pose.

41.3.1.2 Fuse Bank 0

Fuse Bank 0 contains manufacturing information such as Unique ID (UID), consists of fab ID, lot and wafer number, and die coordinates. These bits are usable by OS's such Microsoft PocketPC, which require a unique device ID. It also contains JTAG and debug features controls, IMEI information, and 24 bits for user purpose.

41.3.1.3 Fuse Bank 1

Fuse Bank 1 contains response reference value for the secure JTAG controller, memories information and 40 bits for user purposes.

41.3.1.4 Fuse Bank 2

Fuse Bank 2 contains the 256-bit SCC key.

41.3.1.4.1 SCC Key Format

The-256 bit keys derived from Fuse Bank 2 are used as the secret keys for the Secure RAM controller in the SCC(s). Each key actually consists of three 56-bit keys. Each of the three 56-bit keys corresponds to a 64-bit DES key with the parity bits removed, and processed through the key permutation.

The value in Fuse Bank 1includes 159 random bits and 9 Hamming Code bits. The Hamming Code will detect all 1, 2, and 3 bit errors in the codeword. It will also detect most (though not all) multiple bit errors. Numbering the bits in the key from 0 to 167, the code bits are bits 0, 1, 2, 4, 8, 16, 32, 64, and 128. All of the remaining bits are data bits. Each of the code bits is the modulo 2 sum (i.e XOR or parity) of a subset of the bits in the entire word, as described below.

To determine which bits are used to form each code bit, first look at the binary representation of the bit position of each code bit (ignoring bit 0 for the time being) as shown in Table 41-26.

Table 41-26. Binary Representation of Bit Position

Code Bit	Binary Representation
0	0b00000000
1	0b00000001
2	0b00000010
4	0b00000100
8	0b00001000
16	0b00010000
32	0b00100000
64	0b01000000
128	0b10000000

For code bit number 1, there is a single '1' bit in its binary representation. This code bit is the modulo 2 sum (XOR) of all bit positions which also have a '1' in this same point in their binary representation (that

is, all odd bit positions). Alternatively, if we bitwise AND code bit 1's bit number with a data bit's bit number, and the result is not zero, then that bit gets XOR'ed into code bit 1. The same is true for the other code bits, excepting code bit 0. For example, code bit 2 is the XOR of bits 3, 6, 7, 10, 11, 14, 15 ...254,255. Code bit 128 is the XOR of all bits from 129 through 255 inclusive. After all of the other code bits have been calculated, code bit 0 is simply the XOR of all of the other bits, including the other code bits.

Another way to look at this is to ask which code bits a data bit affects. If we look at the binary representation of the bit position of a data bit, the 1's represent the code bits that this data bit affects. For example, data bit 99 (0b01100011) is XOR'ed into code bits 1, 2, 32, and 64.

41.3.1.4.2 SCC Key Checking

The SCC checks the key using the Hamming Code. If there are any errors in the key, the SCC refuses to use it. In addition, the SCC checks the key against several known weak keys, which it will also refuse to use. Figure 41-23 shows a list of hexadecimal key patterns that are checked (x means don't care); no key matching any of these patterns should be programmed into Fuse Bank 1.

```
00000000000000_xxxxxxxxxxxxxx_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_00000000000000_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_xxxxxxxxxxxxxx_00000000000000
FFFFFFFFFFFFFF_xxxxxxxxxxxxxx_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_FFFFFFFFFFFFFF_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_xxxxxxxxxxxxxx_FFFFFFFFFFFFFF
0000000FFFFFFF_xxxxxxxxxxxxxx_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_0000000FFFFFFF_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_xxxxxxxxxxxxxx_0000000FFFFFFF
FFFFFFF0000000_xxxxxxxxxxxxxx_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_FFFFFFF0000000_xxxxxxxxxxxxxx
xxxxxxxxxxxxxx_xxxxxxxxxxxxxx_FFFFFFF0000000
55555555555555_AAAAAAAAAAAAAA_33333333333333
AAAAAAAAAAAAAA_55555555555555_CCCCCCCCCCCCCC
```

Figure 41-23. Known Weak SCC Key Patterns

41.3.1.5 Fuse Bank 3

Fuse Bank 3 is intended to hold data relevant to the Super-Root Key (SRK) of DSP domain. It may hold an entire SRK (that is, an up to 1024-bit RSA public key), a hash of an SRK (that is, a 160-bit SHA-1 hash), or an SRK one-of-many selector. The boot code for a specific product must know where to find and validate the SRK.

41.3.1.6 Software-Controllable Volatile Signals

The IIM implements up to 28 software-controllable, HW-Visible, volatile control signals. These are implemented in four 8-bit registers, each with a lock bit to inhibit modification of the associated register until the IIM is reset. These 28 signals may all drive SoC-level nets. They are always software-readable,

but can only be modified by software if the Lock bit (bit 7) is not set in the register. The sequence for modifying these bits is shown in Figure 41-24.

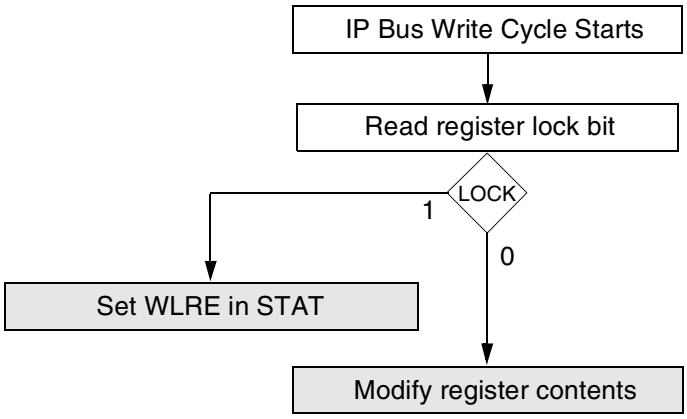


Figure 41-24. SCS Register Write Sequence

41.3.2 Fuse Box Signals

Figure 41-24 shows the fuse box interface.

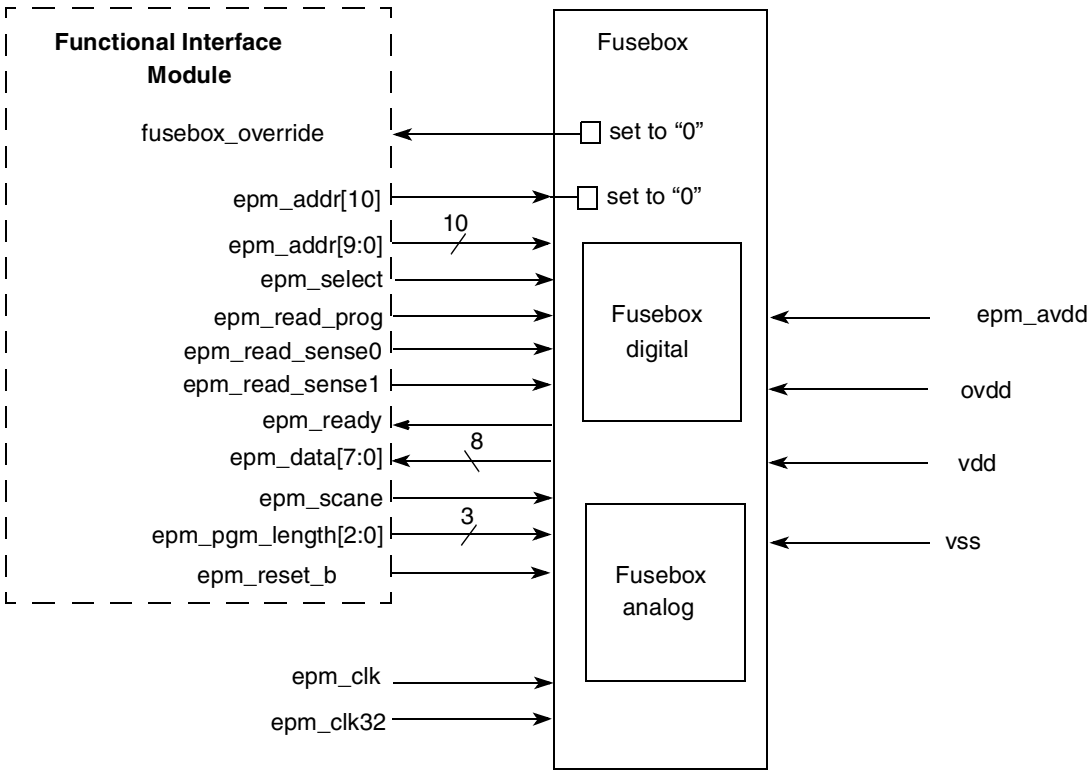


Figure 41-25. Fuse Box Interface

The signals are summarized from the fuse box's perspective in Table 41-27.

Table 41-27. Fuse Box Signal Overview

Signal Name	Signal Type	Signal Description
EPM_ADDR[10:0]	I	This bus provides the address for a read or program operation. Read operations are done on a word (8-bit) basis, so the word address is carried on address[10:3]. Program operations are done on a bit-basis, so the bit address is carried on address[10:0]. Address bits l2:0] carry the bit location in a byte, epm_addr[10] is set to "0" for 1024 bit digital block.
		State Meaning
		Timing Needs to be held to desired value at least 2 nS before and after asserting epm_select in both program and read operation.
EPM_CLK	I	Local clock used for synchronization of signals between FI and Fusebox. It is the clock used by a Fusebox Interface Module for creating Fusebox interface signals. The max. read clock is 166Mhz with worst 40%-60% duty cycle.
EPM_CLK32	I	The 32.768 KHz clock used in the Fusebox to time the program operation in read operation. This clock does not need to be utilized in the FI module.
EPM_DATA[7:0]	O	This bus carries the read data from the Fusebox during a read cycle.
		State Meaning
		Timing
EPM_PGM_LENGTH [2:0]	I	These signals define the length of the program pulse.
		State Meaning
		Timing
EPM_READY	I	This signal indicates the progress of the current read/program cycle. New commands should only be issued by the FI module while epm_ready is asserted.
		State Meaning Asserted -- The fuse is blown. Negated -- The fuse is not blown.
		Timing Latched at the rise edge of epm_clk half clock cycle ahead of epm_ready
EPM_READ_PROG	I	This signal selects whether the current operation is a read or program cycle. This signal is latched into the Fusebox on the rising edge of epm_select.
		State Meaning Assertion— Fusebox is in read operation. Negation— Fusebox is in program operation.
		Timing Needs to be held to desired value at least 2nS before and after asserting epm_select in both program and read operation.

Table 41-27. Fuse Box Signal Overview (continued)

Signal Name	Signal Type	Signal Description
EPM_READ_SENSE0	I	This signal allows sensing of the 0 (unblown) state to be stressed. This can be useful in detecting marginally blown fuses.
		State Meaning Assertion—When asserted (high), the sense circuitry is configured to stress the sensing of the 0 (unblown) state. Negation— No stress is added.
		Timing This signal is latched by the Fusebox on the rising edge of epm_select. It needs to be set to desired value at least 2nS before epm_select is asserted an to be held until epm_select is negated.
EPM_READ_SENSE1	I	This signal allows sensing of the 1 (blown) state to be stressed. This can be useful in detecting marginally blown fuses.
		State Meaning Assertion—When asserted (high), the sense circuitry is configured to stress the sensing of the 1 (blown) state. Negation— No stress is added.
		Timing This signal is latched by the Fusebox on the rising edge of epm_select. It needs to be set to desired value at least 2nS before epm_select is asserted and to be held until epm_select is negated.
$\overline{\text{EPM\_RESET\_B}}$	I	Reset Fusebox digital signals during power up. This signal needs to be connected to a early system reset. The signal is active low.
EPM_SCANE	I	This signal indicates whether the Fusebox is scannable via JTAG. It shall be always de-asserted because the JTAG function is disabled inside Fusebox in cmos065 design.
EPM_SELECT	I	This signal starts a read/program operation and epm_select rising edge needs to be synchronized with the rising edge of epm_clk. There is no restriction for epm_select falling edge to be synchronized with epm_clk. The Fusebox senses the rising edge of this signal to start the operation. On the rising edge of epm_select, the address, operation to perform and program data are latched. There is one select signal for each fuse bank.
		State Meaning Assertion— In Read/program operation. Negation— Not in read/program operation.
		Timing This signal must remain asserted throughout the read/program operation.
FUSEBOX_OVERRIDE	O	Fusebox override signal to the Functional Interface module. It is connected to VSS by default. It can be re-connected to core VDD via Metal 6 layer change.
		State Meaning Assertion— The Fusebox is bypassed. All outputs are set to 0. Negation— The Fusebox outputs are sent to FI modules.
		Timing

41.3.2.1 Fuse Box Operations

All Fusebox operations are self-timed and begin relative to the rising edge on the epm_select line. Most interface signals are shared by all Fuseboxes. Only the Fusebox selects (epm_select), read data (epm_data) and ready status (epm_ready) signals are unique to each Fusebox. The 32 KHz clock is not used within the IIM, but may be used with other FI modules. It is passed through to the Fusebox to time program operations. The 32 KHz clock distributed to the Fuseboxes should be on during Fusebox program operations and should be gated off during read operation. The epm_clk is used to sample the epm_select, epm_ready and epm_data signals in order to synchronize these signals with the Functional Interface inside a Fusebox. This epm_clk is a gated clock used by the local FI modules.

41.3.2.1.1 Word Read

Read operations are done on a word (8-bit) basis. The target word is specified by the address lines epm_addr[9:3]. The bottom three address lines (bit-select lines) are ignored in a read operation.

The timing for fuse word read cycles is shown in Figure 41-26. This timing diagram shows one read cycle followed by the start of a second read, demonstrating how quickly one read can follow another. Each read command is registered on the rising edge of epm_select. The address (epm_addr[9:3]) and read command (epm_read_prog) must be stable tsels before the rising edge of epm_select and must remain stable at least tselh hold time beyond the rising edge. epm_ready is negated immediately after the command is registered, indicating that the Fusebox is busy. After a tdv delay from the active edge of epm_select, the data is made available on blr by the analog Fusebox. The epm_ready acknowledge signal is asserted tready after the initial assertion of epm_select and the data is available on the epm_data. The Fusebox synchronizes epm_ready and the data to the epm_clk clock and drives the synchronized data onto the IP bus. The time between the valid epm_data and epm_ready is signified by tsync in Figure 41-26 and is one-half of one epm_clk cycle. Once the read data has been latched, epm_select is negated by the Functional Interface module. epm_select must remain negated a minimum cycle to cycle delay of tccd before the next cycle (read or program) may begin. epm_data remains at a steady state until replaced by data from a subsequent read cycle.

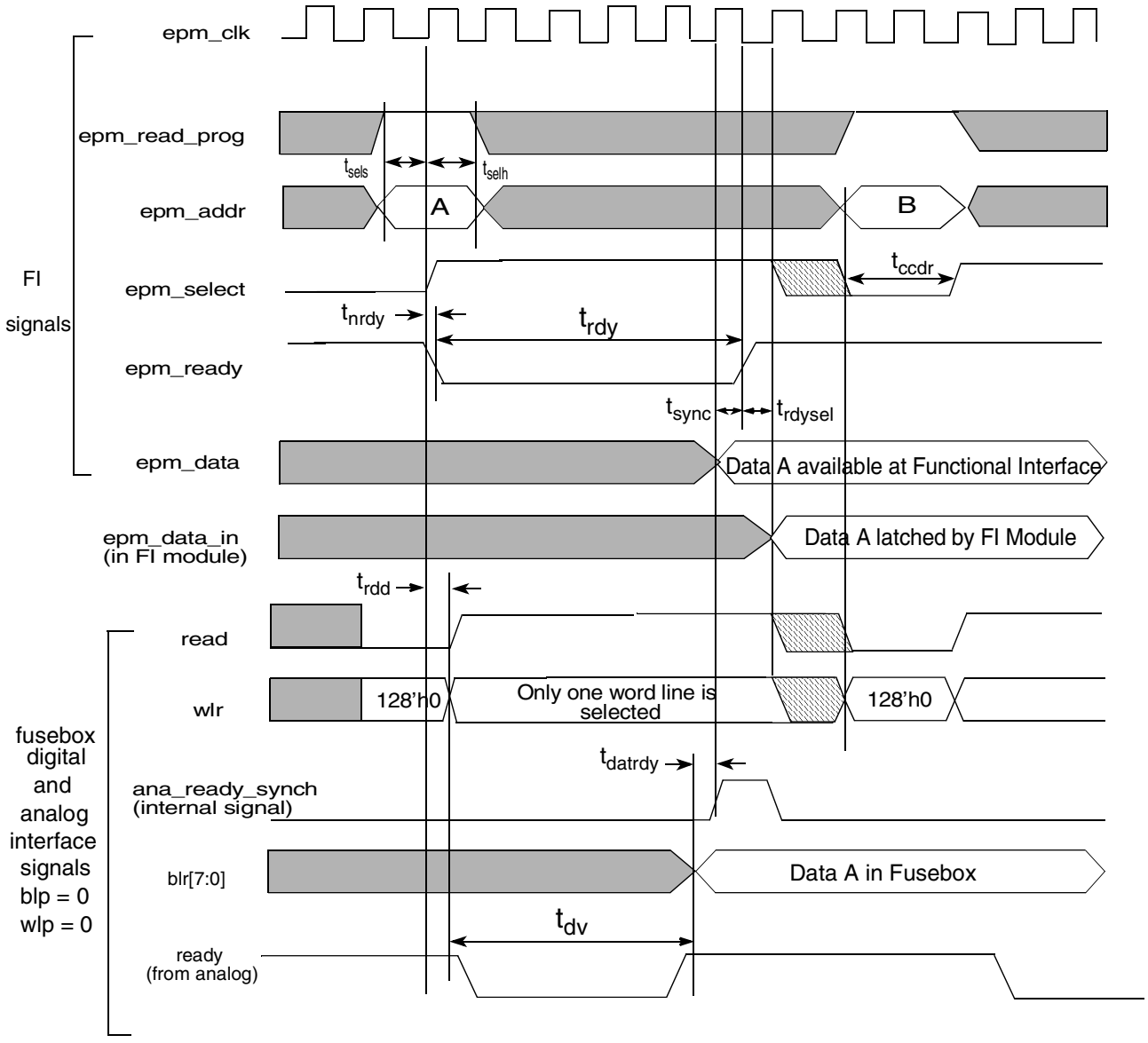


Figure 41-26. Fuse Box Read Cycle Timing

41.3.2.1.2 Bit Program

Program operations are done on a bit-basis. The target bit is specified by the full address bus (epm_addr[10:0]). Program operations can only blow fuses (change them from logic 0 to logic 1), so there is no program data bus associated with the Fusebox. The timing for a program operation is shown in Figure 41-27.

As in the read cycle, the address and program command are latched on the rising edge of epm_select. The address must be setup tsels ahead of the rising edge and remain held tselh past the edge of epm_select. Once the program command is registered, on the next rising edge of the 32 KHz clock, the Fusebox

initiates the program operation. An internal counter starts to count the number of 32K clock cycles, and the epm_ready signal is de-asserted.

After the specified number of 32 KHz clocks (determined by epm_pgm_length[2:0]), the Fusebox completes the program operation and asserts the epm_ready signal. epm_pgm_length[2:0] needs to be set tsels ahead of the rising edge of epm_select and needs to be held until the completion of the internal counter. Consult Table 41-28 for specific timing values. The Functional Interface module synchronizes the epm_ready response to the epm_clk during the period tsync. After recognizing that the program operation has completed, the Functional Interface module negates the epm_select line. To avoid accidental selection of more than one bit line at a time, the 8 bit de-coded bit-line-program (BLP) needs to be checked before generate blp[7:0].

Table 41-28. Timing Values

Designator	Timing Value	Description
t_{sels}	2 nS	Setup time to epm_select active edge
t_{selh}	2 nS	Hold time after epm_select active edge
t_{nrdy}	200 pS –1000 pS	Delay from epm_select to epm_ready negation (read). Delay from the rise edge of 32 Khz clock to epm_ready (program).
t_{rdd}	1 nS–4 nS	Assert time from epm_select active edge to negation of read from digital
t_{dv}	30 nS–100 nS	Data valid after epm_select active edge
t_{rdy}	≤ (t_{dv} + 1.5 epm_clk)	Epm_ready signal asserted after epm_select active edge (read only)
t_{ready}	(epm_pgm_length) * $period_{32KHz}$	Epm_ready signal asserted after epm_select active edge (program only)
t_{datrdy}	0–1 epm_clk	Data driven out to bus after analog ready signal asserts (read only)
t_{sync}	one-half of 1 epm_clk cycle (rising edge to falling edge) > 3nS if epm_clk duty cycle is not 50%	epm_data to epm_ready synchronization delay (read). epm_ready to negation of epm_select (program)
t_{rdysel}	> 2nS typically one-half of 1 epm_clk cycle (falling edge to rising edge)	rising-edge of epm_ready to negative-edge of epm_select (read only)
t_{ccdr}	5 nS	Minimum cycle to cycle delay (Consecutive Read)
t_{ccdp}	0.5 uS	Minimum cycle to cycle delay (Consecutive program)
t_{sels}	2 nS	Setup time to epm_select active edge
t_{selh}	2 nS	Hold time after epm_select active edge
t_{nrdy}	200 pS –1000 pS	Delay from epm_select to epm_ready negation (read). Delay from the rise edge of 32 Khz clock to epm_ready (program).

See the timing diagram of the fuse program in Figure 41-27.

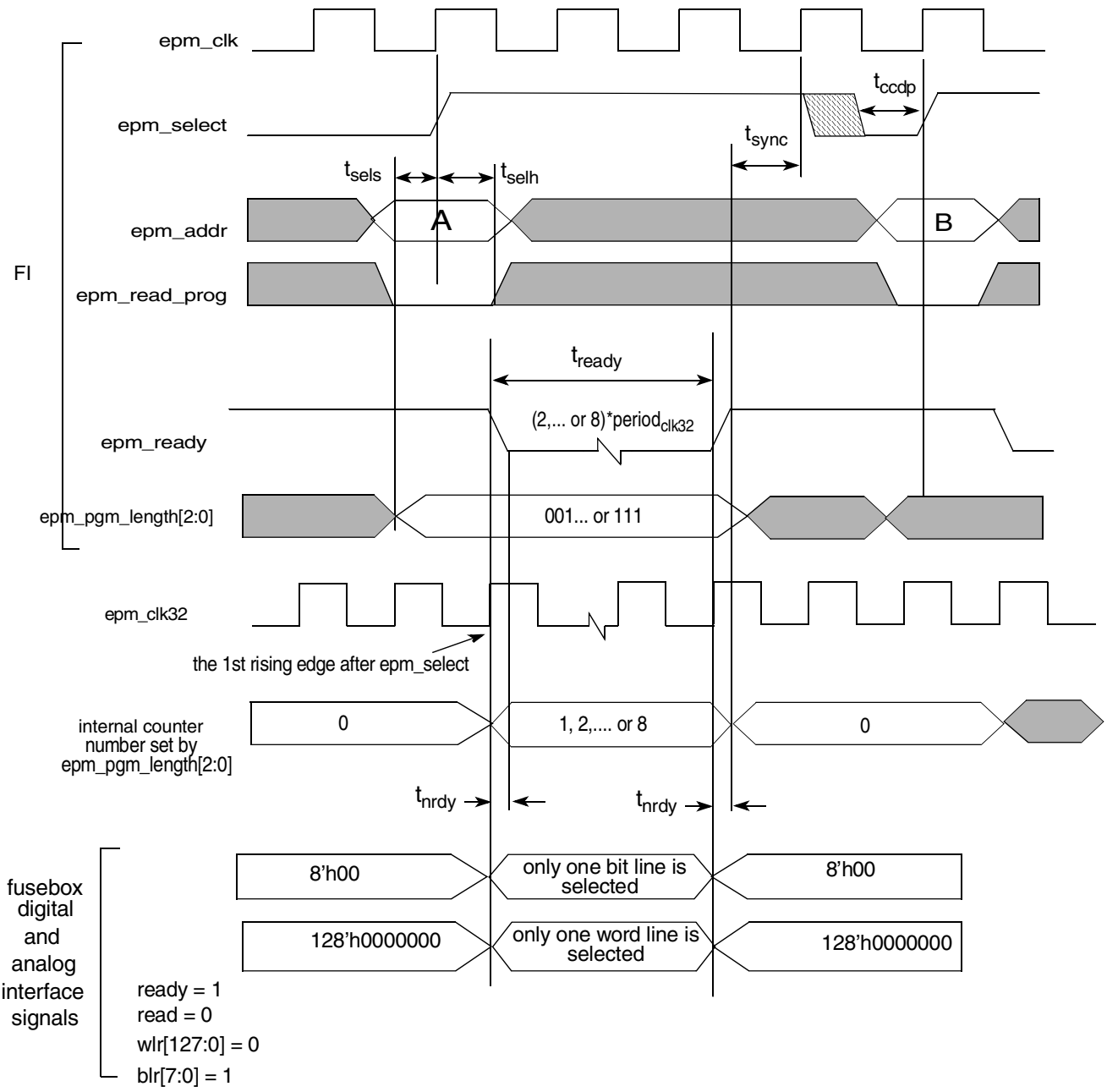


Figure 41-27. Fuse Program Cycle Timing

NOTE

The IIM controller must determine that the target fuse bank is indeed an e-Fuse bank (not a laser fuse bank) before allowing the program cycle to proceed. This is determined by examining the appropriate fuse_type signal, which is fixed at SoC integration time.

41.3.3 Fuse Value Storage

The values of fuses may be read from one of the following three places:

- Software fuse value shadow cache
- Hardware-visible fuse value shadow cache, driving SOC nets
- Fuse elements themselves

The reasons for caching the fuse values are to reduce the risk of accidental programming of e-Fuses due to repeated reads and to reduce power consumption associated with sense cycles.

41.3.3.1 Software Fuse Value Shadow Cache

Each word in the cache RAM includes a valid bit and a parity bit, as shown in Figure 41-28. The valid bit is set if the data bits and parity bit have been set according to the sensed state of the corresponding fuses, Odd parity is used across the data bits only (that is, the valid bit is not included in the parity calculation).

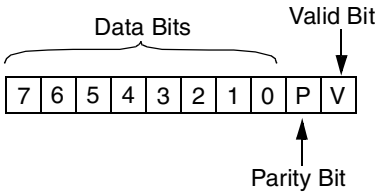


Figure 41-28. Software Cache Word Format

All bits are set to 0 when IIM comes out of reset. When a fuse word is read, the IIM will first attempt to read it from the cache (except for the hardware-visible fuses; see Section 41.3.3.2, Hardware-Visible Fuse Shadow Cache"). A sense cycle will only be run to the fuses if the cached word is invalid or the parity is incorrect. The overall cached fuse word read sequence is shown in Section 41.3.5.1, Read Sequence." The cache words and fuse words is one to one mapped.

41.3.3.2 Hardware-Visible Fuse Shadow Cache

The hardware-visible fuse include fuses in bank0 and bank1, and FBAC words of each bank.The IIM must sense these fuses and write their values to the appropriate registers when it comes out of reset, and must ensure than any change to the fuses is also immediately reflected in the registers. The overall hardware-visible word read sequence is shown in Section 41.3.5.1, Read Sequence." Each word has a parity bit as shown in Figure 41-29.

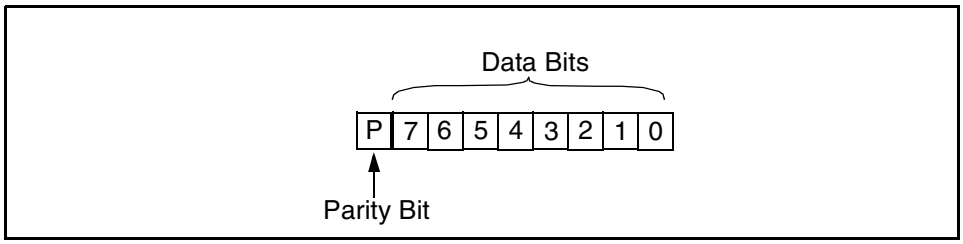


Figure 41-29. Hardware Cache Word Format

If parity error detected on shadow cache, IIM will set error bit PARITYE, and re initialize cache from fuses.

41.3.4 Fuse Protection

There are some fuses used as protection bit.

41.3.4.1 Fuse Box Bank Protection Fuse

FBWP of each bank control whether this bank can be programed. FBOP of each bank control whether this bank can be overridden. FBRP of each bank control whether this bank can be read. FBSP control whether this bank can be scanned. FBESP control whether this bank can be explicitly sensed.

41.3.4.2 Word Lock Bit

Refer to Chapter 6, “Fuse Map,” for word locks, the type of protection they provide, and words they lock.

41.3.4.3 Scan Out Protection

To prevent the secret keys scanned out by hacker, specific logic is added (shown in Figure 41-30). It resets all flip-flops in IIM before entering scan mode.

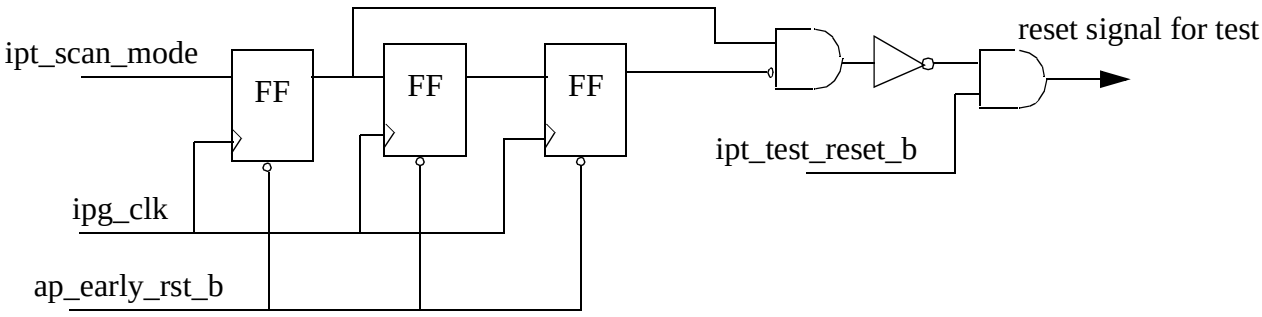


Figure 41-30. Scan Out Protection Circuit

41.3.5 Fuse Bank Operations

This section discusses the following sequences:

- Section 41.3.5.1, Read Sequence”
- Section 41.3.5.2, Explicit Sense Sequence”
- Section 41.3.5.3, Programming Sequence”
- Section 41.3.5.4, Override Sequence”

41.3.5.1 Read Sequence

Fuses may be read using the IP bus interface from any bank which is not read-inhibited (that is, FBACx[FBRP] is unblown). The read sequence from a hardware-visible signals word is shown in Figure 41-31.

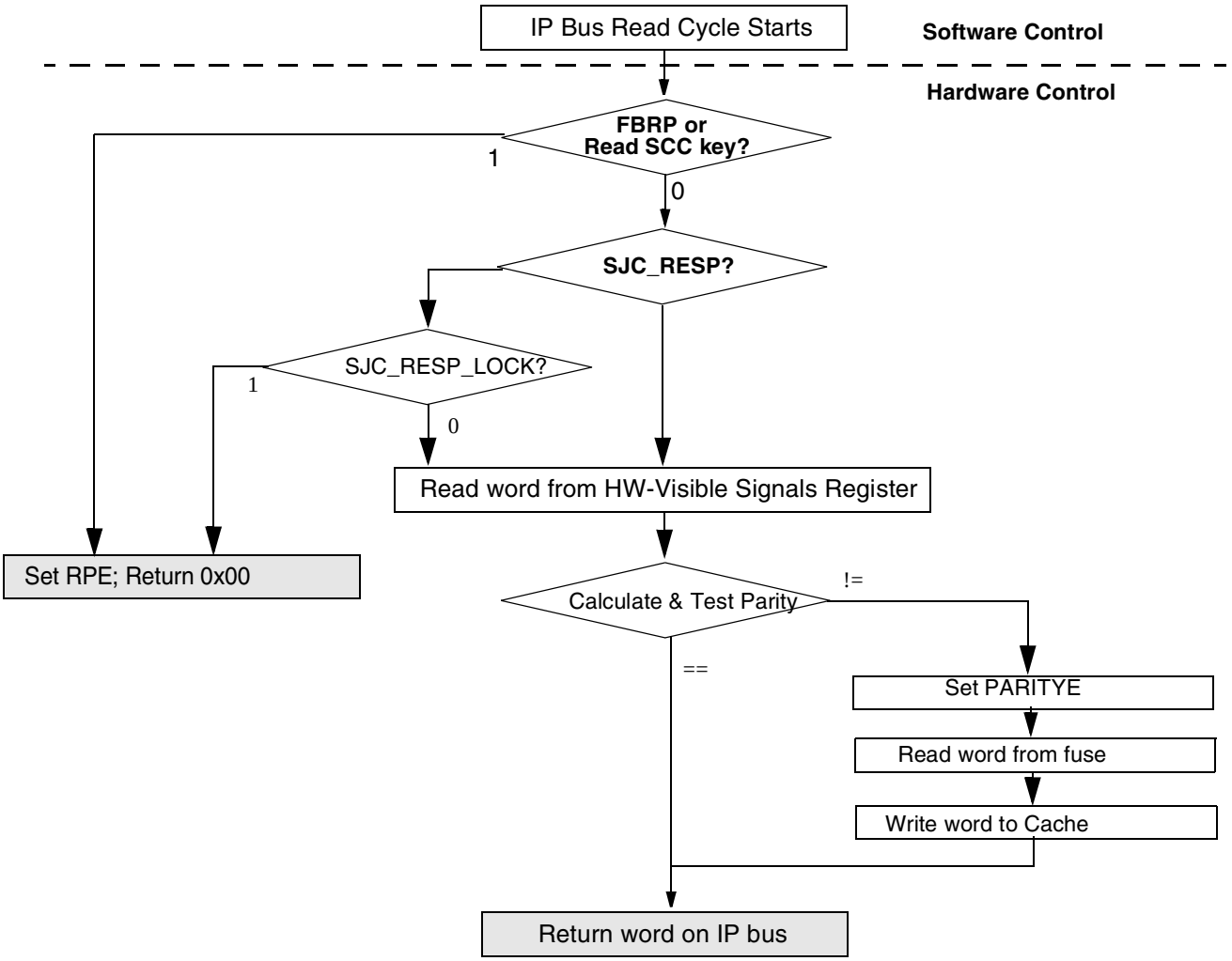


Figure 41-31. Hardware Visible Fuse Read

The read sequence to a cacheable fuse word is shown in Figure 41-32.

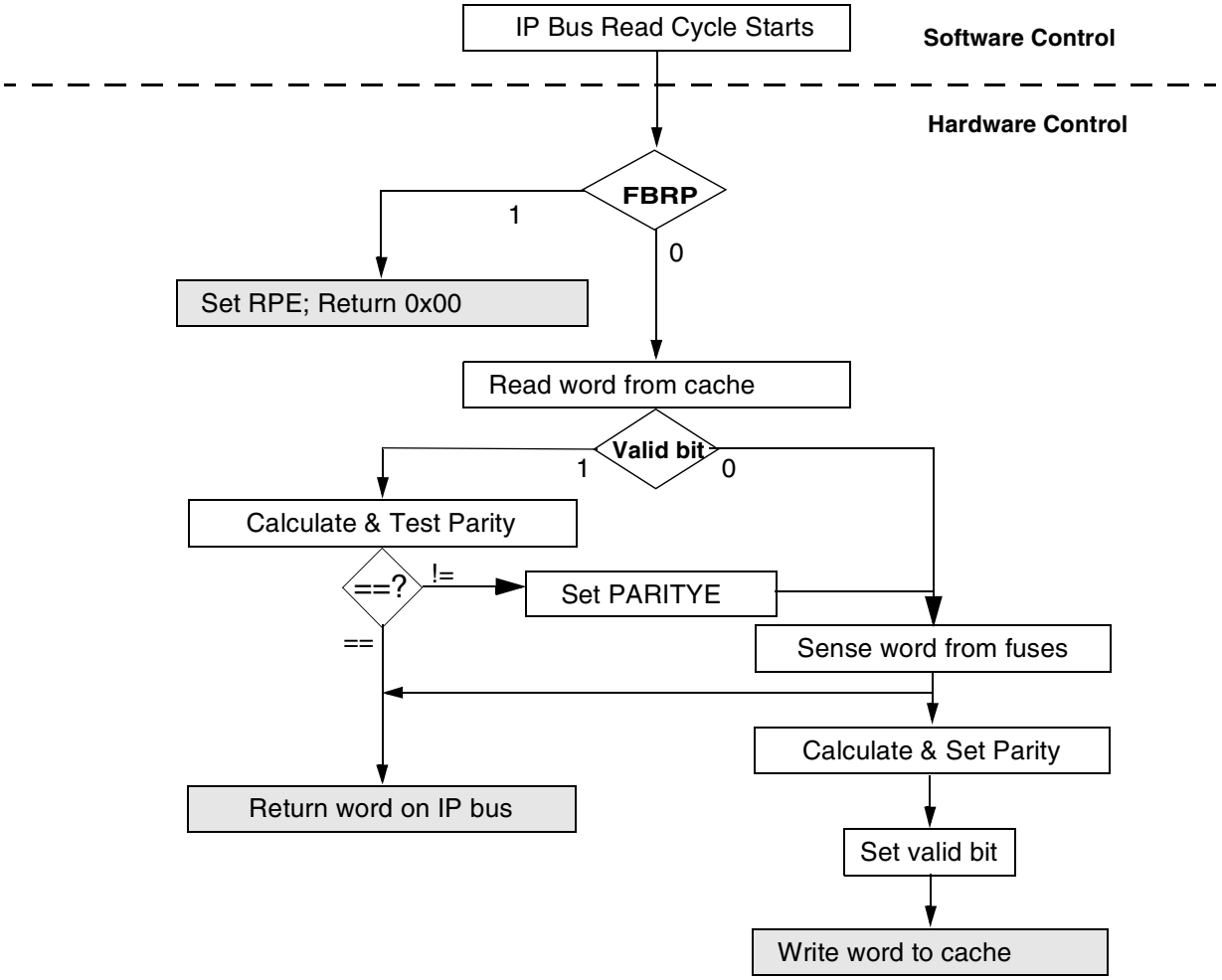


Figure 41-32. Software Fuse Read

41.3.5.2 Explicit Sense Sequence

The explicit sense sequence using IP bus is depicted in Figure 41-33.

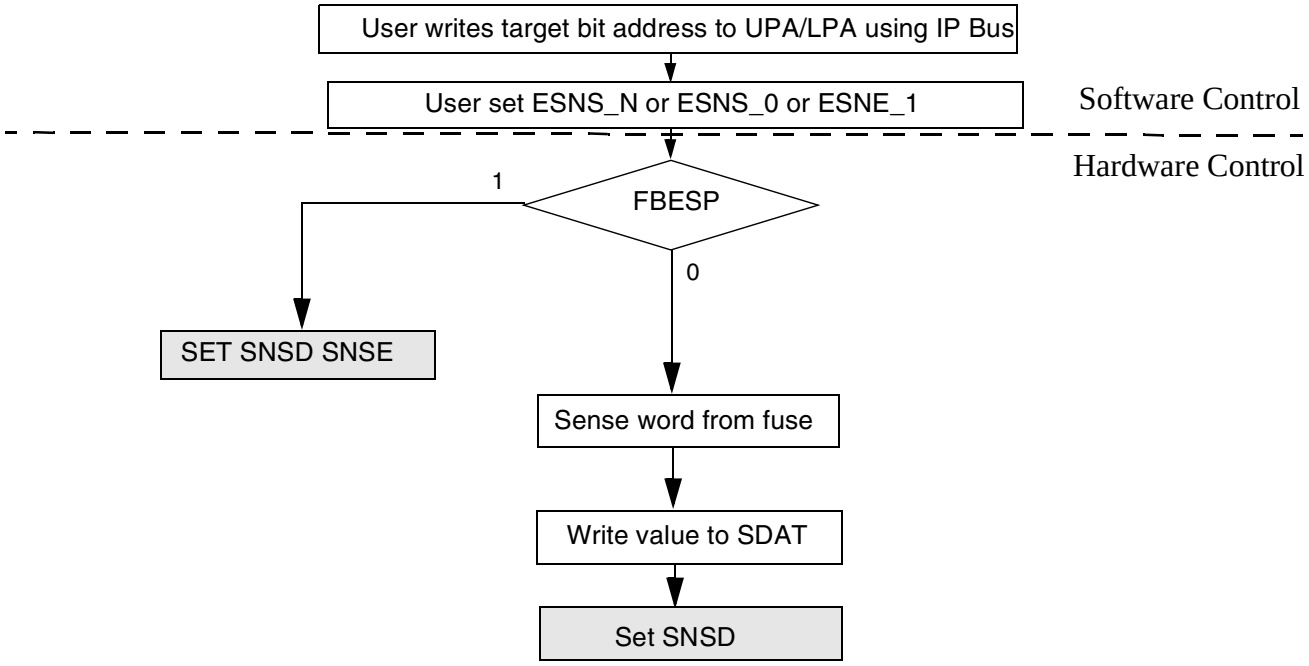


Figure 41-33. Explicit Sense Sequence

41.3.5.3 Programming Sequence

The software-controlled e-Fuse programming sequence using the IP bus is depicted in Figure 41-34.

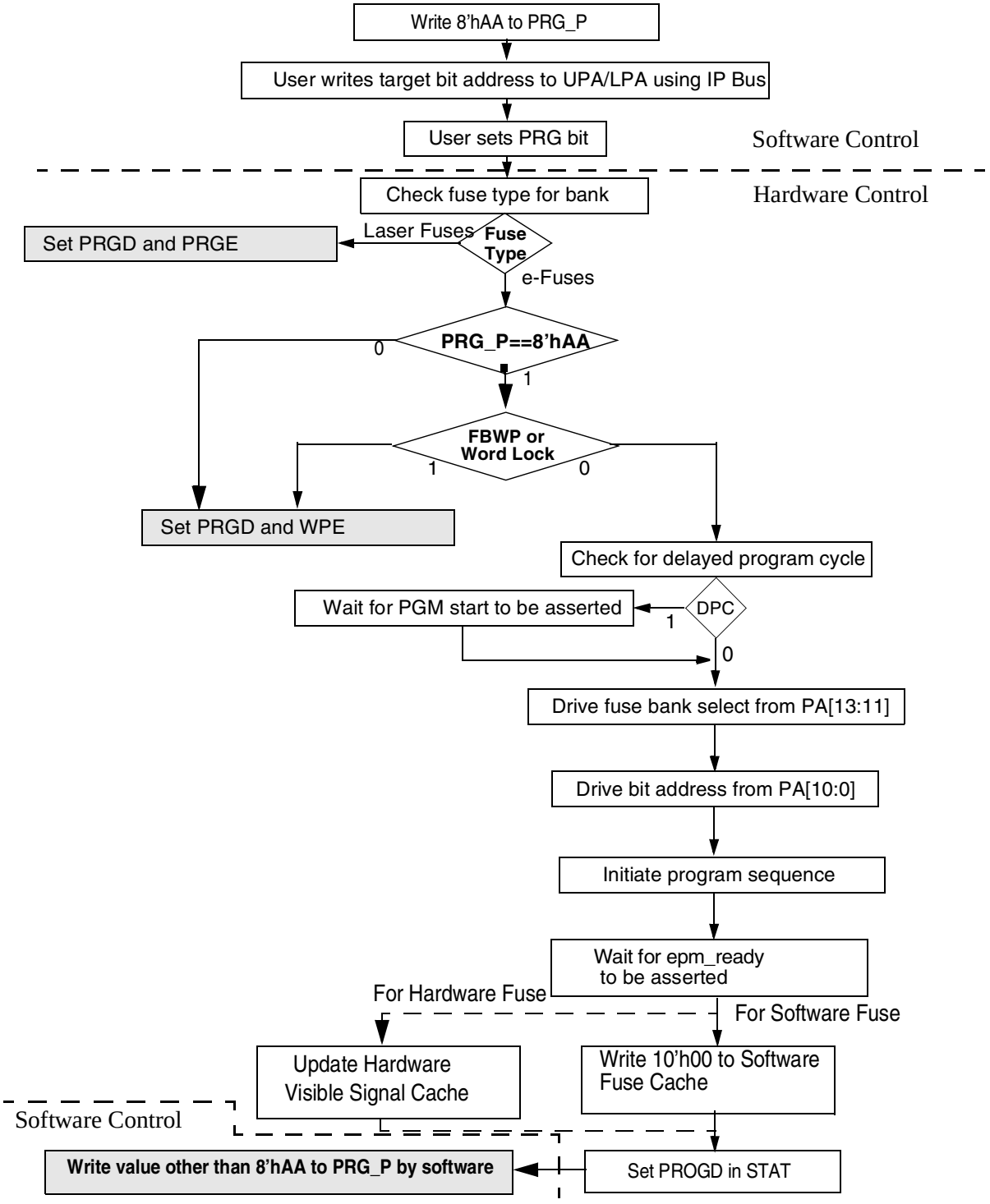


Figure 41-34. e-Fuse Program Sequence

Fuses may also be programmed directly using the JTAG interface, so long as the IIM asserts the appropriate epm_scane signal. The IIM is not involved in this programming sequence aside from allowing or disallowing it using the epm_scane signal.

Note that minimum program voltage is 2.775 V, maximum program voltage is 3.3 V. This must be taken into account when designing the architecture of an IC which includes e-Fuses. For complete parametric specifications, refer to the fuse box specification.

41.3.5.4 Override Sequence

The override sequence is depicted in Figure 41-35.

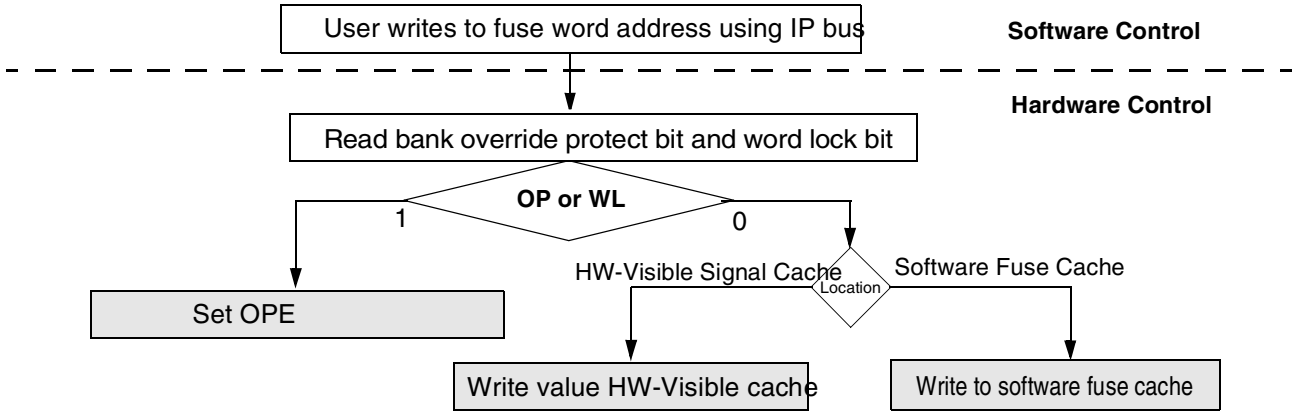


Figure 41-35. Fuse Value Override Sequence

41.4 Initialization/Application Information

41.4.1 Initialization

When the IIM come out of reset, IIM automatically senses the hardware-visible fuses and writes their values into the appropriate registers. IIM also senses the fuse access control word from each of the fuse banks, and set epm_scane according to the values of fbsp bits. Software fuses are not sensed until software has made a read request for them. Software cache will be set to 0 on reset.

After the IIM come out of reset, all the hardware visible fuses must be read out to hardware visible cache within 200us, to ensure the hardware visible fuses are loaded before the second reset at system level. IIM reads HWV2 first. iim_boot_int_ready is asserted after this read to indicate hardware visible fuse value of BOOT_INT is valid. Then FBAC words of all banks are sensed out, the sequence of sensing of other fuses depends on the parameter definition, which will be elaborated in Section 41.4.3, Parameter Definition.” The initialization time depends on the number of hardware visible fuses.

Initialization time = (number of hardware visible fuse word) × (100 ns + 4 ipg_clk cycles)

After the initialization is complete, fuse_latched will be asserted, as shown in Figure 41-36.

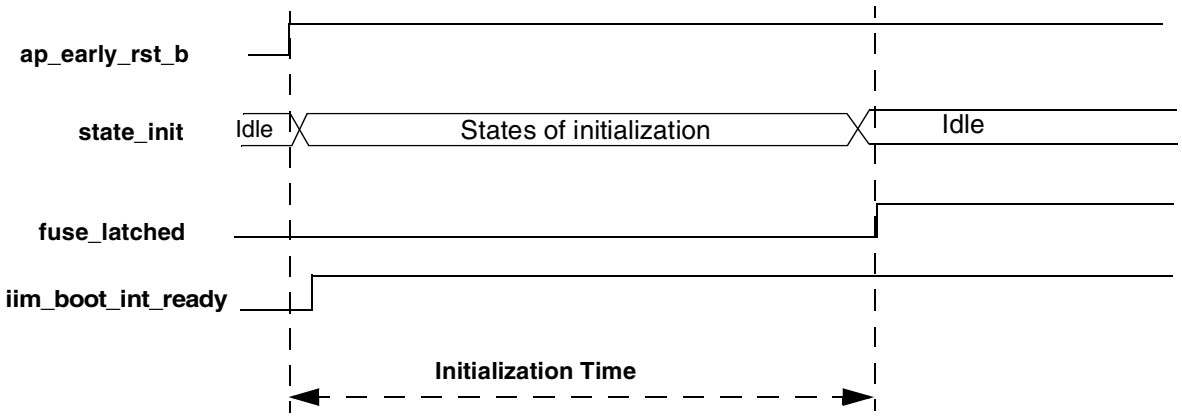


Figure 41-36. Timing Diagram of Initialization

41.4.2 Program

After JATG programming, reset must be applied to synchronize the fuse value to hardware visible signal cache.

A wait period is required between program and read operation.

41.4.3 Parameter Definition

This section for the hardware design. Each project should generate parameter file according to this section.

The RTL design can be parameterized. The maximum number of fuse bank is 8. Maximum usable fuse bank size are 2048 bits. All the parameter are defined in file iim_para. The following parameters need to be defined.

41.4.3.1 Parameter for Number

The parameters for number are as follows:

- num_bank defines the number of fuse bank.
- num_ram_block defines the number of blocks of software fuses.
- num_ram defines the number of words in software fuses cache.
- num_hwv_block defines the number of blocks of hardware visible fuses.
- num_hwv defines the number of words in hardware visible fuses cache.

41.4.3.2 Parameter for Address

This section discusses the parameter for address.

41.4.3.2.1 Address Boundary of Fuse Bank

al_bankx defines the low boundary of fuse bank x. au_bankx defines the up boundary of fuse bank x. The maximum number of fuse bank is 8. Assign 0 to unused banks.

Example 41-1. Address Boundary of Fuse Bank

```
parameter  [31:0]  num_bank= 5;
parameter [13:0] al_bank0=14'h0800;
parameter [13:0] au_bank0=14'h087C;
parameter [13:0] al_bank1=14'h0c00;
parameter [13:0] au_bank1=14'h0c3C;
parameter [13:0] al_bank2=14'h1000;
parameter [13:0] au_bank2=14'h107C;
parameter [13:0] al_bank3=14'h1400;
parameter [13:0] au_bank3=14'h147C;
parameter [13:0] al_bank4=14'h4;
parameter [13:0] au_bank4=14'h0;
parameter [13:0] al_bank5=14'h4;
parameter [13:0] au_bank5=14'h0;
parameter [13:0] al_bank6=14'h4;
parameter [13:0] au_bank6=14'h0;
parameter [13:0] al_bank7=14'h4;
parameter [13:0] au_bank7=14'h0;
```

41.4.3.2.2 Address Boundary of Software Fuse Block

al_ram_blockx defines the low boundary of software fuse block x. au_ram_blockx defines the up boundary of software fuse block x. The fuse block which is defined in this section will be mapped to Software Fuse Cache.The maximum number of blocks is 16. Assign 0 to unused blocks. If there is no fuse to be mapped to Software Fuse Cache, num_ram should be defined to 1 to avoid Verilog syntax error.

Example 41-2. Address Boundary of Software Fuse Block

```
parameter [31:0] num_ram = 32'd31;
parameter [31:0] ram_addr_width = 5;
parameter [13:0] al_ram_block0=14'h1404;
parameter [13:0] au_ram_block0=14'h147C;
parameter [13:0] al_ram_block1=14'h4;
parameter [13:0] au_ram_block1=14'h0;
```

```
parameter [13:0] al_ram_block2=14’h0;
parameter [13:0] au_ram_block2=14’h0;
parameter [13:0] al_ram_block3=14’h0;
```

41.4.3.2.3 Address Boundary of Hardware Visible Signal Block

al_hwv_blockx defines the low boundary of hardware visible Signal block x. au_hwv_blockx defines the up boundary of hardware visible fuse block x. The fuse block which is defined in this section will be mapped to Hardware Visible Signal Cache. The maximum number of blocks is 16. Assign 0 to unused block. Please note SCS0~SCS3 are assigned in block0

Example 41-3. Address Boundary of hardware Visible Signal Block

```
parameter [31:0] num_hwv = 32’d86;
parameter [31:0] hwv_addr_width = 7;
parameter [13:0] al_hwv_block0=14’h002C;
parameter [13:0] au_hwv_block0=14’h0038;
parameter [13:0] al_hwv_block1=14’h0800;
parameter [13:0] au_hwv_block1=14’h0800;
parameter [13:0] al_hwv_block2=14’h0c00;
parameter [13:0] au_hwv_block2=14’h0c00;
parameter [13:0] al_hwv_block3=14’h1000;
parameter [13:0] au_hwv_block3=14’h1000;
parameter [13:0] al_hwv_block4=14’h1400;
parameter [13:0] au_hwv_block4=14’h1400;
parameter [13:0] al_hwv_block5=14’h0;
parameter [13:0] au_hwv_block5=14’h0;
parameter [13:0] al_hwv_block6=14’h0844;
parameter [13:0] au_hwv_block6=14’h0844;
parameter [13:0] al_hwv_block7=14’h0804;
parameter [13:0] au_hwv_block7=14’h0840;
parameter [13:0] al_hwv_block8=14’h0848;
parameter [13:0] au_hwv_block8=14’h087c;
parameter [13:0] al_hwv_block9=14’h0c04;
parameter [13:0] au_hwv_block9=14’h0c3c;
parameter [13:0] al_hwv_block10=14’h1004;
parameter [13:0] au_hwv_block10=14’h107c;
parameter [13:0] al_hwv_block11=14’h0;
parameter [13:0] au_hwv_block11=14’h0;
parameter [13:0] al_hwv_block12=14’h0;
parameter [13:0] au_hwv_block12=14’h0;
parameter [13:0] al_hwv_block13=14’h0;
parameter [13:0] au_hwv_block13=14’h0;
parameter [13:0] al_hwv_block14=14’h0;
parameter [13:0] au_hwv_block14=14’h0;
```

The sequence of the definition defined the sequence of initialization. When IIM come out of reset, it initialize block1 first, then block2, block3, and so on. The sequence of the parameter definition is the same as the sequence of the fuse in hardware visible signal cache. But please note that changing of the sequence of blocks which belong to first two banks need some changes in RTL(iim_hwv_reg.v). FBAC words of all banks must be placed together.

In ZATs, Fuses in Bank0, Bank1 and FBAC words of all banks are forced to be Hardware Visible Fuses, so they must be defined in hardware visible signal block. that is, al_hwv_blockx, au_hwv_blockx.

41.4.3.3 Other Parameters

The other parameters are as follows:

- chl_size defines the size of sjc_challenge, resp_size defines the size of sjc_response.
- hwv_addr_width defines the width of hardware visible signal cache.
- ram_addr_width defines the width of software fuse cache.

If the size of bank0 is larger than 128 fuses(16 words), the following statement should be added in the iim_para.

```
‘define bank0_para_size
‘define bank0_size_para
```

Chapter 42
Image Processing Unit 3 (IPUv3EX)

42.1 Introduction

42.1.1 Overview

The goal of the IPUv3EX is to provide comprehensive support for the flow of data from an image sensor and/or to a display device. This support covers all aspects of these activities:

- Connectivity to relevant devices - cameras, displays, graphics accelerators, TV encoders and decoders.
- Related image processing and manipulation: sensor image signal processing, display processing, image conversions, etc.
- Synchronization and control capabilities (for example, to avoid tearing artifacts)

This integrative approach leads to several significant advantages:

- Automation:
  The involvement of the MCU (Main Control Unit) in image management is minimized. In particular, display refresh/update and a camera preview (displaying the input from an image sensor) can be performed completely autonomously. The resulting benefits are reducing the overhead due to SW-HW synchronization, freeing the MCU to perform other tasks and reduced power consumption (when the MCU is idle and can be powered down).
- Optimal data path:
  Access to system memory is minimized. In particular, significant processing can be performed on-the-fly while receiving data from an image sensor and/or sending data to a display. System memory is used essentially only when a change in pixel order or frame rate is needed. The resulting benefits are reduced load on the system bus and further reduction of power consumption.
- Resource sharing:
  Maximal HW reuse for different applications, resulting with the support of a wide range of requirements with minimal HW

The HW reuse mentioned above is enabled by a sophisticated configurability of each HW block. This configurability also allows the support of a wide range of external devices, data formats and operation modes. The resulting flexibility is important also because the support requirements are evolving significantly, so expected future changes need to be anticipated and accounted for.

The following further principles guided the choice of support provided by the IPUv3EX:

- For key applications that DESERVE and NEED HW support (for acceleration or low power), provide the best support (leading to an optimal implementation)
- For additional applications that can benefit from the HW, consider cost vs. benefit of making minor modifications/extensions to support them
- For all other relevant applications (to be supported by SW), verify that their support is not degraded.
- Whenever possible, let the operating system (and its windowing system) act as it would without the IPUv3EX.

42.1.2 Architecture

A simplified block diagram of the IPUv3EX is given in Figure 42-1. The role of each block is described in Table 42-1.

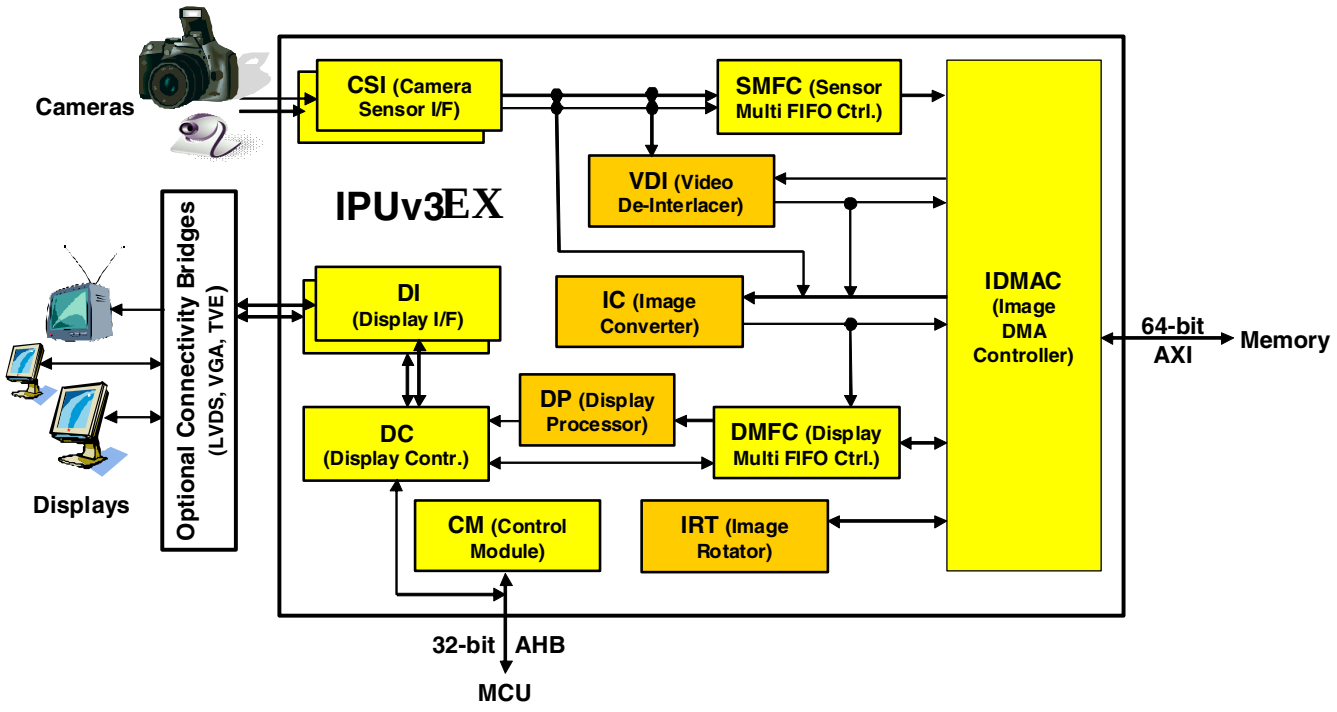


Figure 42-1. IPUv3EX - Simplified Block Diagram

Table 42-1. IPUv3EX - Block Description

Block	Description
CSI - Camera Sensor Interface	Controls a camera port; provides interface to an image sensor or a related device. IPUv3EX includes 2 such blocks
DI - Display Interface	Provides interface to displays, display controllers and related devices. IPUv3EX includes 2 such blocks
DC - Display Controller	Controls the display ports.

Table 42-1. IPUv3EX - Block Description

Block	Description
DP - Display Processor	Performs the processing required for data sent to display
IC - Image Converter	Performs resizing, color conversion/correction, combining with graphics, and horizontal inversion
VDI - Video De Interlacer	Performs video de interlacing (Interlaced -> progressive)
IRT - Image Rotator	Performs rotation (90 or 180 degrees) and inversion (vertical/horizontal)
IDMAC - Image DMA Controller	Controls the memory port; transfers data to/from system memory
SMFC - Sensor Multi FIFO Controller	Controls FIFO's for output from the CSI's to system memory
DMFC - Display Multi FIFO Controller	Controls FIFO's for IDMAC channels related to the display system
CM - Control Module	Provides control and synchronization.

42.1.3 Features And Functionality

42.1.3.1 External Ports

The IPUv3EX has the following ports:

- Two camera ports - each controlled by a CSI module - providing a connection to image sensors and related devices.
- Two display ports - each controlled by a DI module - providing a connection to displays and related devices.
- Memory port - AXI (AHB V3.0) master, controlled by the IDMAC - providing connection to the system memory.
- AHB-lite slave port, providing connection to the ARM MCU (and to any other master connected to the ARM's cross-bar switch)
- Additional ports for control and debug

42.1.3.1.1 Camera Ports

The role of these ports is to receive input from image sensors (or TV decoders) and to provide support for time-sensitive control signals to the camera. (non-time-sensitive controls - for example, configuration, reset - are performed by the MCU, through an I2C I/F or GPIO signals).

Each of the camera ports includes the following features:

- Direct connectivity to most relevant image sensors and to TV decoders.
- Interface types
  - Parallel interface
    - Up to 20-bit input data bus.
    - A single value in each cycle, except for special cases listed in Table 42-2 (comments column)
    - Programmable polarity

- The data formats
  - Interleaved color components, up to 16 bits per value (component)
  - The supported formats are listed in Table 42-2.

Table 42-2. Data Formats Supported By The Camera Port

Format	Resolution	On-The-Fly Processing	Direct path to memory	Comments
Bayer RGB	8 bits/value	Yes	Yes	
	9-10 bits/value	Yes, starting with companding to 8 bits	Written to the MSB of a 16-bit word	
	11-16 bits/value	Yes, for parallel I/F only starting with companding/reduction to 8 bits		
Full RGB or YUV 4:4:4	444/555 mode	Yes, starting with color extension to 8 bits/sample	Yes	In parallel I/F: through an 8-bit or 16-bit bus
	565 mode			In parallel I/F: through an 8-bit or 16-bit bus
	8 bits/value (888 mode)	Yes	Yes	
	8-16 bits/value	Yes, starting with companding/reduction to 8 bits	8- or 16 bit components are written to the MSB of a 16-bit word 10 bits/value can also be packed in a 32-bit word	
YUV 4:2:2 Component order: UY1VY2... or Y1UY2V...	8 bits/value	Yes	Yes	In parallel I/F: through an 8-bit bus (e.g, BT.656) or 16-bit bus (for example, BT.1120)
	10 bits/value	Yes - for parallel I/F only; starting with companding to 8 bits	Written to the MSB of a 16-bit word	In parallel I/F: through a 10-bit bus (for example, BT.656) or 20-bit bus (for example, BT.1120)
	11-16 bits/value	Yes, starting with reduction to 8 bits	Written to the MSB of a 16-bit word	
Gray scale	8 bits/value	Yes	Yes	
	9-16 bits/value	Yes, starting with companding/reduction to 8 bits	Written to the MSB of a 16-bit word	
Generic data		No	Yes In a parallel I/F, if wider than 8 bits, each bus word is written to the MSB of a 16-bit word	May be used for any other formant, for example, JPEG/MPEG4

- Scan order: progressive or interlaced
  Interlaced data (expected only for YUV 4:2:2) is sent directly to system memory, from where it can be read back for further processing.
- Frame size: up to 8192 x 4096 pixels

- Synchronization - video mode
  - The sensor is the master of the pixel clock (PIXCLK) & synchronization signals
  - Synchronization signals are received using either of the following methods:
    - Dedicated control signals -VSYNC, HSYNC - with programmable pulse width & polarity
    - Controls embedded in the data stream, following loosely the BT.656 protocol , with flexibility in code values and location.
- Synchronization - still image capture
  - The image capture is triggered by the MCU or by an external signal (for example, a mechanical shutter)
  - Synchronized strobes are generated for up to 6 outputs - the sensor and camera peripherals (flash, mechanical shutter...)
- Additional features
  - Frame rate reduction, by periodic skipping of frames
    - The supported reduction ratios are: m:n, where m,n<=5
    - This is supported independently for the different destinations - IC, SMFC.
  - Window-of-interest selection
  - Color depth reduction
    - From up to 16 bits down to 10 bits: by truncating LSBs
    - From 10 bits to 8 bits: companding - programmable piecewise-linear map (may incorporate color gain corrections)
  - Pre-flash - for red-eye reduction and for measurements (for example, focus) in low-light conditions

Several sensors can be connected to each of the CSI's. Simultaneous functionality (sending data) is supported as follows:

- Two sensors can send data independently, each through a different port.
- Only one of the (non-generic) streams can be transferred to the , VDI or IC for on-the-fly processing, while the other ones are sent directly to system memory.

The input rate supported by the camera port is as follows:

- Parallel interface:
  - Average: up to 90M values/sec
    Bayer - 90 M pixels/sec (for example, 6M pixels @ 15 fps)
    YUV 4:2:2 - 45 M pixels/sec (for example, 3M pixels @ 15 fps)
    YUV 4:4:4 or RGB - 30 M pixels/sec (for example, 3M pixels @ 10 fps)
  - Peak: up to 120M values/sec
    (to account for up to 35% blanking intervals)
- Fast serial interface:
  - Average: up to 120M values/sec, 1.44 Gbps
    Bayer - 120M pixels/sec (for example, 8M pixels @ 15 fps, 12 bits per value)

YUV 4:2:2 - 60M pixels/sec (for example, 4M pixels @ 15 fps, 12 bits per value)
YUV 4:4:4 or RGB - 40M pixels/sec (e.g., 4 pixels @ 10 fps, 12 bits per value)

  - Peak: up to 160 values/sec, 1.92 Gbps
    (to account for up to 35% blanking intervals)
- When two (or more) sensors are used simultaneously, the total rate supported is as above.
- On-the-fly processing (as listed in Table 42-2) may be restricted to a lower input rate - see Section 42.1.3.2 for further details.
- I/O pads may also be restricted to a lower input rate

42.1.3.1.2 Display Ports

The role of these ports is to communicate with display devices, either directly or through a controller (e.g. graphics accelerator) or a bridge (e.g. TV encoder).

Access Modes

Two access modes are supported.

Synchronous access:

In this mode, the IPUv3EX transfers a two-dimensional block of pixels to the display device, in synchronization with the screen refresh cycle.

This mode has a dual role:

- For a RAM-less display or a TV screen, this mode is used to perform the screen refresh process from a display buffer in system memory.
- For a “smart” display, this mode is used to transfer a rectangular block of pixels to the display’s screen and, in some cases, also to the display buffer
  - The transferred block may be only part of the screen (the rest of the screen being refreshed by the integrated controller, from the internal buffer). Moreover, a mask can be used to transfer to the display only parts of the block, e.g., a window partly hidden by other windows.
  - If the block is transferred only to the screen, the transfer rate must be equal to the refresh rate. If, however, the transfer is also to the display’s memory, the rate can be reduced to the rate at which the input buffer is updated.

In all cases (including the last one), the IPUv3EX sends to the display all the synchronization signals controlling the screen refresh and the block transfer is synchronized with these signals. This synchronization means that tearing effects are avoided when using this mode.

Asynchronous Access:

This is the main mode used for communicating with an external display controller (possibly in a smart display or a graphics accelerator). In this mode, the IPUv3EX performs random access - read/write - to the memory and registers of the controller.

Two types of addressing methods are supported

- Generic linear addressing of pixels and generic data

- 2-dimensional (X/Y) addressing of pixels

The following access types are provided:

- Data transfer to the external device, after on-the-fly processing in the IPUv3EX.
- Data transfer (DMA) - read/write - between the host's system memory and the external device, through the IPUv3EX's memory port (controlled by the IDMAC); e.g. transfer of a rectangular block of pixels (possibly full screen).
- Host access - read/write - to an external device, through the AHB-slave port
  - Access types
    - Direct access - emulating a directly-addressed access (see below)
      This includes burst access (incremental; up to 8 words/burst)
    - Low-level access - leaving to the host the explicit generation of the access protocol
  - The possible accessing modules include the MCU and the system DMA controller (as well as any other AHB master connected to the MCU's cross-bar switch).

Transfer of video/graphics data stream to controller's display buffer is performed using one of the first two modes above. Unlike in the synchronous mode, this process is not tightly-synchronized with the screen refresh cycle. However, a loose synchronization - to avoid tearing - is still possible: the appropriate timing for the transfer can be derived from the VSYNC signal of the screen refresh - either generated by the IPUv3EX's display controller or received from the external controller.

The asynchronous access requires the specification of an address. The display interface uses "indirect addressing", namely, there is no address bus, and the address, as well as control and configuration commands, are embedded in the data stream. The access procedure - including writing addresses and commands - is managed autonomously by the interface, in one of two ways:

- Automatic emulation of transparent access, following microcode ("access template") generated by the MCU. This mechanism is very flexible, supporting a wide variety of devices.
- Streaming commands/addresses from a buffer stored (by the MCU) in system memory.

Note that direct access requires the use of the first method - automatic emulation.

The Interface

The display interface is very flexible and supports a wide variety of devices from major manufacturers. The following interface types are provided (in each of the two display ports)

- Parallel video interface (for synchronous access) - up to 24-bit data bus.
  - Control protocol - follows Sharp HR and generic TFT definitions
  - Supports BT.656 (8-bit) and BT.1120 (16-bit) protocols
  - Supports HDTV standards SMPTE274 (1080i/p) and SMPTE296 (720p)
- A parallel bidirectional bus interface (for asynchronous access) - up to 32-bit data bus.
  - Compatible with MIPI-DBI standard .
  - Control protocol - either system-80 or system-68K
    The timing and polarity of the signals are programmable.
  - Byte-enable - optional, for a 16-bit interface

— Burst access
For direct access, the burst is determined by the corresponding signal in the AHB interface.

- A serial interface - 3-wire, 4-wire and 5-wire (two flavors) (for asynchronous access)

The supported formats for pixel data are

- RGB - color depth fully configurable; up to 8 bits/value (color component)
- YUV 4:2:2, 8 bits/value (for TV encoder)

In the parallel interfaces, the data bus has up to 32 bits. Non-trivial mapping of pixels to the bus is restricted to the 24 LSB's. This mapping is fully configurable and very flexible. In the serial interfaces, the data is mapped in the same way as in the parallel interfaces and then serialized.

The interface also supports "generic data". Such data is transferred - byte-by-byte, without modification - between the system memory and the display device (through a serial interface or 8/16-bit parallel interface). Non-conventional pixel formats can be supported by considering them as "generic data".

For the interface clock, there are the following options (independently for each port)

- Derived from the IPUv3EX internal clock (master mode)
- Provided by an external source (slave mode)

The transfer rate supported:

- For single port (for all interfaces):
  - 110 Mega-Accesses/Sec if the DI clock is derived from an external to IPUv3EX source (like another PLL)
  - 120 Mega-Accesses/Sec if the DI clock is derived from the IPUv3EX clock (HSP_CLK)

For synchronous access with one cycle/pixel, this enables, e.g (including 35% blanking intervals)

— XGA (1024x768) @ 100 fps
— 720p (1280x720) @ 60 fps
— 1080i (1920x1080) @ 30 fps
—

- The combined rate for the two ports is up to 120 MP/sec

The interface includes the following additional features:

- Screen size: up to 4096 x 2048 pixels, programmable by software.
- Scan Order: progressive or interlaced
- Synchronization
  — Programmable horizontal and vertical synchronization output signals (for synchronous access)
  — Data enabling output signal
- Software contrast control using 8-bit programmable pulse-width modulation (PWM)
  Two dedicated PWM outputs are provided

Connecting To Display Devices

IPUv3EX allows the connectivity to multiple display devices. In particular, it supports the following setup:

- Primary LCD display; can be smart, dumb (RAM-less) or dual-port; may use the parallel interface.
- Second LCD display; can be smart or dumb (RAM-less); may use parallel or serial interface .
- TV Output: either digital parallel output, or (through an integrated TV encoder bridge) analog output..

Each of the above connections has independent settings - interface timing, access template, chip-select, etc.

Simultaneous functionality of the above devices is possible in each of the following ways:

- Two devices can be accessed (synchronously or asynchronously) independently, each through a different port: each using any of the available interfaces.
- Two devices can time-share asynchronous accesses through the legacy serial & parallel interfaces, using the CS signals.
- An asynchronous access can be performed during vertical blanking intervals of a synchronous access (screen refresh; to the same or other device).

The possibilities for simultaneous functionality by time-sharing the legacy interfaces in a single port are summarized in Table 42-3

Table 42-3. Simultaneous Functionality of Display Port By Time-Sharing Legacy Interfaces

	Second/Third Display Type		
Primary Display Type	**Smart Display Serial Interface**	**Smart Display Parallel Interface Asynchronous access**	**RAM-less display or TV screen**
Smart Display Parallel Interface Asynchronous access	Yes	Yes	Yes; access to the smart display is restricted to blanking intervals
Dual Port Smart Display Synchronous access	Yes	Yes, access to the secondary display is restricted to blanking intervals	Not Available
TFT RAM-less Display Or TV screen	Yes	Yes, access to the smart display is restricted to blanking intervals	Not Available
Graphics Accelerator	Yes	Yes, if the accelerator supports a chip select functionality	Not Available

42.1.3.1.3 Memory Port

The memory port is an AXI (AHB V3.0) master port, used to read/write data - typically two-dimensional blocks from/to system memory.

The interface supports the following features

- Clock rate up to 133 MHz (equal to the internal clock)
- 64-bit data bus

- The supported data formats are listed in Table 42-4.

Table 42-4. Data Formats Supported By The Memory Interface

Format	Resolution	Input/Output	Comments
Non-interleaved YUV (in three separate buffers)	8 bits/value	both	4:4:4, 4:2:2, 4:2:0 formats
Partially-interleaved YUV (in two separate buffers)	8 bits/value	both	4:2:2, 4:2:0 formats Y buffer and UV buffer
Interleaved YUV (all color components in a single buffer)	8 bits/value	both	4:4:4 format (YUV...) 4:2:2 format (UY1VY2... or UY2VY1... or Y1UY2V...or Y2UY1V...)
Interleaved true color	8, 12,16, 18, 24, 32 bits/pixel 0 - 8 bits per R/G/B/A value	both	Flexible component location A: translucency value (only in input)
Coded color (using a palette)	4,8 bits/pixel	input only	
Gray scale	8 bits/pixel	both	
	4 bits/pixel	input only	Transferred to display port
Generic data (Transparent M)	8 bits/unit	both	E.g.: From CSI; to DI; Compressed data to/from DP Translucency for combining

  - The pixel formats are translated to/from a uniform internal format: RGBA/YUVA 8:8:8:8
- The supported ordering of bytes and pixels is little endian. For 4 bits/pixel, big endian is also supported.
- Addressing modes include:
  - Sequential access (to a contiguous memory buffer) - for generic data.
  - Raster-scan within a two-dimensional window of a video/display buffer - for both pixel and generic data.
  - Raster-scan of two-dimensional blocks within a two-dimensional window (for rotation of pixel data)
- Additional features
  - Scan order: progressive or interlaced
    Interlaced access is supported for fields which are stored either in separate memory buffers or with rows interleaved in a single buffer.
  - Reordered scan, implementing inversion and rotation.
    - Rotation and horizontal inversion - only when transferring two-dimensional blocks (to/from the IRT)
    - Vertical inversion - also in row-by-row raster-scan
  - Scrolling

– Applications
Panning within a frame
Frame scrolling
– Not supported for non-interleaved and partially-interleaved formats
– Resolution
Vertical: single pixel
Horizontal: 18 BPP - 4 pixels; 12, 4 BPP or YUV 422 - 2 pixels; other formats - one pixel

— Conditional read (for combining): fully-transparent or hidden pixels are not read.
– This is supported, for graphics. by reading the transparency (alpha) from a separate buffer

— Input/output FIFOs (in the SMFC, DMFC and in the processing modules) - size adjusted to provide resilience for latency of up to 1500 cycles.

42.1.3.2 Processing

The IPUv3EX processes rectangular blocks of pixels. The processing is performed in these modules - , VDI, DP, IC and IRT (see Figure 42-1 and Table 42-1).

42.1.3.2.1 Processing flows

Several time-shared data flows are supported, as described in Table 42-5.

Table 42-5. Time-Shared Data Flows Through The IPUv3EX

Name	Number	Type	Flow	Target	Restrictions
Display Refresh/ Update	5 flows (at most two of them of type DS1)	DS1	Fmem -> DP -> Display	Synchronous Access (e.g. display refresh; controlled by the DI)	
		DS2	Fmem -> DP -> Display	Asynchronous Access (e.g. display update)	
		DS3	Fmem <-> Display	Generic Data Transfer	
	1 flow	DS4	MCU <-> Display	Direct Access	
Video Playback	3 flows	PL1	Bmem -> VDI -> IC -> Bmem -> IRT -> Fmem + DSx	Main option	
		PL2	Fmem -> IRT -> Bmem -> IC -> DP	Low power (branching to DSx, as a video plane)	Large enough window No other video flows
		PL3	Fmem -> VDI -> IC -> DP	Low power (branching to DSx, as a video plane)	Interlaced source Large enough window No other video flows

Table 42-5. Time-Shared Data Flows Through The IPUv3EX

Name	Number	Type	Flow	Target	Restrictions
Camera Pre-view	2 flows (VF2 may be used also as a playback flow)	VF1	Sensor -> IC -> Bmem -> IRT -> Fmem + DSx	main option	Single progressive input
		VF2	Sensor -> Fmem -> /VDI -> IC -> Bmem -> IRT -> Fmem + DSx	two inputs and/or interlaced input	When the VDI is used, one of the three input fields can go directly from the sensor to the VDI. In that case the sensor output goes to the memory via the VDI and not the SMFC.
		VF3	Sensor -> IC -> DP	Low power Smart Display Tearing-less (branching to DS2, as a video plane)	Single progressive input Refresh rate = 2x sensor frame rate Large enough window No other video flows
		VF4	Sensor -> IC -> Fmem + DS1	Low power RAM-less Display Single Display Buffer (in internal memory) Tearing-less	Single progressive input Refresh rate = 2x sensor frame rate Large enough window No other video flows
Video Record	2 flows	RCx	IC -> Bmem -> IRT -> Fmem	(branching from VFx)	
Graphic Overlays	2 flows	GF1	Fmem -> IC	(combining with the main flow	
	2 flow	GF2	Fmem -> DP		

Comments

- System memory usage - legend
  - Fmem: frame double-buffer (page-flip) in system memory (typically external)
  - Bmem: two possibilities
    - A frame double buffer, as above
    - A band (4-256 rows) double-buffer (page-flip) in system memory (could be internal)
  - Direct arrow between two processing stages represents an internal pipeline
- Time-sharing
  - IC can time-share tightly three flows: one VFx, one RCx and one PLx (with independent processing parameters)
    –
  - DP can time-share one DS1 flow and a one DS2 flow (each with different destinations and independent processing parameters)
  - Direct access to display (DS4) time-shares tightly the display port with other active DSx flows.
  - Other time-sharing (between PLx; between DS2&DS3; in IRT) is frame-by-frame

- Any of the processing stages in the above flows can be skipped.
- Triggering and synchronization
  - Flow segments starting from a sensor are triggered and synchronized by input from the sensor
  - Flow segments ending with display refresh are triggered and synchronized by the refresh control mechanism in the DI.
  - Flow segments starting and ending in system memory, are triggered either by the double buffering mechanism or by explicit configuration and are processed continuously without delay, at a rate determined by the available resources. These flow segments have a lower priority than the sensor/display-driven flows.

The functionality of each of the processing blocks is described below.

42.1.3.2.2 Display Processor (DP)

The Display Processor performs all the processing required for data sent to a display.

- Input: from the IC and/or from system memory
  - Order: rows, progressive or interlaced
  - Format: YUVA/RGBA, non-decimated, 8 bits/value
- Processing chain
  - Combining 2 video/graphics planes
  - Overlaying a simple HW cursor
    32 x 32 pixels, uniform color; may be combined logically with the full plane.
  - Color conversion/correction - linear (multiplicative & additive)
    Programmable; including:
    - YUV <-> RGB, YUV<->YUV conversions
      where YUV stands for any one of the color formats defined in the MPEG-4 standard
    - Adjustments: brightness, contrast, color saturation...
    - Special effects: gray-scale, color inversion, sephia, blue-tone...
    - Color-preserving clipping, for gamut mapping
    - Hue-preserving gamut mapping - for minimal color distortion
    - Applied to the output of combining or to one of the inputs
  - Gamma correction and contrast stretching - programmable piecewise-linear map
- Output: to display (through the DC)
  - Rate: up to 120M pixels/sec (e.g., WXGA (1366x768) @ 85 fps + 35% blanking intervals)
  - Format: YUV/RGB, non-decimated, 8 bits/value

The DP processes a single data flow at any given time, but it supports up to three data flows, by time sharing

- A Primary flow:
  - The input is loaded periodically - using a timer (as required, e.g. for a synchronous access)

— Optionally, frames are skipped if the content has not changed (as appropriate for a smart display)

- Two secondary flows:
  — Asynchronous; processed when the DP is not needed for the primary flow (during blanking intervals or when a primary frame is skipped).
  — The two secondary flows are switched frame-by-frame.

42.1.3.3 Video De-Interlacer (VDI)

The Video De-Interlacer converts an interlaced video stream to progressive order, using a high-quality 3-field motion-adaptive filter.

- Video source - SDTV: 480i30 (720x480 @ 30 fps) or 576i25 (720x576 @ 25 fps)
- Input: - three consecutive fields
  — Source
    – The most recent field may come from the CSI or from system memory
    – The other two fields are read from memory
  — Field size: up to 720x512 pixels
  — Pixel format: YUV 4:2:2/4:2:0, 8 bits/value
- Output: progressive frame
  — Destination: to system memory or to the Image Converter.
  — Frame size: up to 720x1024 pixels
  — Rate: up to 60 MP/sec (e.g., 720x576 @ 120 fps + 10% vertical blanking overhead)
  — Format: same as input format

The de-interlacing is performed using a high-quality 3-field filter which is motion adaptive:

- For slow motion - retains the full resolution (of both top and bottom fields)
- For fast motion - prevents motion artifacts

The VDI supports a single video stream at any given time.

42.1.3.3.1 Image Converter (IC)

The Image Converter performs various operations on a video stream.

- Input: from sensor or from system memory
  — Frame size: up to 4096x4096 pixels
  — Rate: up to 100M pixels/sec (e.g., 2.5 MP @ 30 fps + 35% blanking intervals)
  — Order: rows, progressive.
  If resizing is not used, interlaced order is also acceptable.
  — Pixel format: YUV/RGB, 8 bits/value
- Processing chain:
  — Resizing

  - Fully flexible resizing ratio
    Maximal downsizing ratio: 16:1
    Subject to this limitation, any N->M resizing can be performed
  - Independent horizontal and vertical resizing ratios.
- Color conversion/correction - linear (multiplicative & additive)
  Programmable; including:
  - YUV <-> RGB, YUV<->YUV conversions
    where YUV stands for any one of the color formats defined in the MPEG-4 standard
  - Adjustments: brightness, contrast, color saturation...
  - Special effects: gray-scale, color inversion, sephia, blue-tone...
- Combining with a graphics plane (e.g. application-specific overlay)
- Horizontal inversion

- Output: to system memory or (for a single active flow) to a display device (through the DP).
  - Frame size: up to 1024x1024 pixels
  - Rate: up to 50M pixels/sec (e.g., SVGA (800x600) @ 75 fps + 35% blanking intervals)
  - Rate: up to 50M pixels/sec (e.g., SVGA (800x600) @ 75 fps + 35% blanking intervals)
  - Order: rows, progressive.
    If resizing is not used, interlaced order is also acceptable.
  - Format: YUV/RGB, 8 bits/value

The IC supports three time-shared data flows: record, camera preview and playback (the first two share a common input).

42.1.3.3.2 Image Rotator (IRT)

- Input/output: from/to system memory
  - Rate: up to 50M pixels/sec
  - Order: raster scan of 8x8-pixel blocks
  - Format: YUV/RGB, non-decimated, 8 bits/value
- Transformation: combination of the following
  - 90-degree rotation
  - Horizontal inversion
  - Vertical inversion

42.1.3.4 Automatic Procedures

The IPUv3EX is equipped with powerful control and synchronization capabilities to perform its tasks with minimal involvement of ARM and minimal use of memory. In particular, it includes:

- An integrated DMA controller with an AXI master port, allowing autonomous access to the system memory.
- An integrated display controller, performing screen refresh of a RAM-less display.

- A page-flip double buffering mechanism, synchronizing read and write access to system memory, to prevent tearing effects.
- A double/triple buffer synchronization mechanism with a video/graphics source.
- Internal synchronization, e.g., between input from sensor and output to display

As a result, in most cases, the MCU is involved only when it also performs part of the processing (e.g. video coding). In particular, the following procedures are performed by the IPUv3EX completely autonomously:

- Screen refresh for RAM-less displays
- Update of the ("partial plane") display buffer used for screen refresh (located either in system memory or in an external display controller, e.g. of a smart display or graphics accelerator), when the content is generated in a different ("full plane") buffer.
- Camera preview (displaying a view finder - a video-stream from the image sensor to the display) is autonomous except for minimum MCU involvement (~one time per frame or less) in camera input control (exposure, white balance, focus and flicker reduction)

Typically, there are extended periods of times in which there is no other activity in the system. The MCU - being idle - can be put to a low-power mode, reducing the power consumption and extending significantly the battery life.

The IPUv3EX supports several techniques to reduce further the power consumption of the display system:

- Dynamically optimized screen refresh rate (see Section 42.1.3.4.1 below)
- Optimized update of the display buffer (see Section 42.1.3.4.2 below)
- Dynamic backlight control, with low-light compensation by image enhancement

Further features and capabilities of the automatic procedures are outlined below

42.1.3.4.1 Screen Refresh

- The refresh rate may vary within a predefined range. Within this range, the rate is dynamically adjusted to the content update rate.

  An indication about the availability of new content is obtained as follows

  — If the page-flip double buffering is used, the mechanism provides this indication

  — If only a single buffer is used (and incrementally updated), the IPUv3EX can receive an indication of a modification, either through a dedicated external "snooping" signal (e.g. from the EMI), or from the MCU (by setting an internal flag).
- The IPUv3EX counts the refresh cycles: the total and those with new content. The MCU can use these counters to optimize display management (e.g. switching display buffer compression on/off). The counters are reset by the MCU.
- The transferred data may be processed on the way, using the IC and DP.

42.1.3.4.2 Update Of The Display Buffer

- Conditional update
  The IPUv3EX can receive an external "snooping" signal indicating a modification of the full plane

buffer (as during screen refresh above). It monitors the signal and, upon detection, it performs one of the following:

— Performs an update, without any SW intervention

— Interrupts the MCU, that can initiate some more involved procedure (e.g. selective update)

- Automatic display of a changing image (animation) or moving image (scrolling)
  This is implemented by reading frames (from a full plane buffer) with incremental offset. When the IPUv3EX reaches the last programmed frame, it can perform one of the following:
  — Return to the first frame, without any SW intervention
  — Interrupt the MCU, to generate the next content.
- The timing of the update can be adjusted to avoid tearing.
- The transferred data may be processed on the way, using the IC and DP

42.1.3.4.3 Camera Preview

- Tearing artifacts can be prevented by (automatic) page-flip double buffering in system memory
- Alternatively, the video stream from an image sensor can be sent directly to the display buffer used for screen refresh. The significance of this option is that only a single frame buffer is needed (and not two). This buffer may be located either in system memory or in an external display controller.
  — This option is useful, e.g., in a low frame rate, when tearing is not visible.
  — When tearing must be prevented, the refresh cycle in the display can be synchronized with the timing signals from the sensor (two refresh cycles for each input frame): the IPUv3EX receives a VSYNC signal from the sensor and generates from it synchronization signals for the display.

42.1.4 .Modes of Operation

42.1.4.1 Normal Operating Modes

42.1.4.2 Low Power Modes

See 42.3.12.10, “Low Power Modes - Stop, PG and LPSR modes

42.1.4.3 Debug Mode

42.2 Memory Map and Register Definition

The address space for accesses through the AHB-lite slave port is 512 MB and it is split internally (with 32MB resolution) according to bits [28:25] of the address. Using the following notation

Address = (IPU_ID[31:29], MSB[28:25], LSB[24:0]) ***Eqn. 42-1***

the address is used as follows ("T" is a configurable integer between 0 and 13 by configuring the MCU_T parameter):

1. MSB<T: access to an external device, with address = (MSB, LSB)
2. T<=MSB<14: access to an external device, with address (MSB-T, LSB)
3. MSB=14: Low-level access to an external device, with LSB[3:0] = (Lock, CS, RS[1:0])
   - LSB[5:4] = RS[1:0] (the address on the display interface)
   - LSB[6] = Choice of display's channel (0=channel 8, 1=channel 9)
   - LSB[7] = Lock (Lock=1 prevents the use of the display port until the next MCU access)
4. MSB=15: access to internal IPUv3EX registers, with address LSB

The display port has 2 channels dedicated for MCU accesses via the AHB bus. Channel 8 is dedicated to the accesses where MSB<T. Channel 9 is dedicated to the accesses where MSB>T. When only a single channel is used, channel 8 should be used. The usage of channel 9 requires the usage of channel 8 as well.

The table below shows the register file of the module (addresses 0x1E000000 to 0x1FFFFFFF) and how the memory space allocation for each sub module of the IPUv3.

NOTE

The addresses given in Table 42-5 are relative to the IPUv3 base address defined at SoC's level.

Table 42-1. IPUv3EX internal memory map

Start address	End Address	size (Kbytes)	Module	Comment
0xBASE+0x00000000	0xBASE+0xDFFFFFF	30720	2 X Display's Devices	This memory region can be split between 2 display devices connected to a single or separate DIs. The partition between them is defined by the MCU_T parameter.
0xBASE+0xE000000	0xBASE+0xE007FFF	32	CM - Control modules registers	
0xBASE+0xE008000	0xBASE+0xE00FFFF	32	IDMAC	

Table 42-1. IPUv3EX internal memory map

Start address	End Address	size (Kbytes)	Module	Comment
0xBASE+0xE010000	0xBASE+0xE017FFF	32		
0xBASE+0xE018000	0xBASE+0xE01FFFF	32	DP	The DP registers cannot be accessed directly, the user should access these registers via the SRM.
0xBASE+0xE020000	0xBASE+0xE027FFF	32	IC	
0xBASE+0xE028000	0xBASE+0xE02FFFF	32	IRT	
0xBASE+0xE030000	0xBASE+0xE037FFF	32	CSI0	
0xBASE+0xE038000	0xBASE+0xE03FFFF	32	CSI1	
0xBASE+0xE040000	0xBASE+0xE047FFF	32	DI0	
0xBASE+0xE048000	0xBASE+0xE04FFFF	32	DI1	
0xBASE+0xE050000	0xBASE+0xE057FFF	32	SMFC	
0xBASE+0xE058000	0xBASE+0xE05FFFF	32	DC	
0xBASE+0xE060000	0xBASE+0xE067FFF	32	DMFC	
0xBASE+0xE068000	0xBASE+0xE06FFFF	32	VDI	
0xBASE+0xE070000	0xBASE+0xEFFFFFF	15936	RESERVED	
0xBASE+0xF000000	0xBASE+0xF01FFFF	128	CPMEM	
0xBASE+0xF020000	0xBASE+0xF03FFFF	128	LUT	

Table 42-1. IPUv3EX internal memory map

Start address	End Address	size (Kbytes)	Module	Comment
0xBASE+0xF040000	0xBASE+0xF05FFFF	128	CM Shadow (SRM)	
0xBASE+0xF060000	0xBASE+0xF07FFFF	128	TPM	
0xBASE+0xF080000	0xBASE+0xF09FFFF	128	DC template	
0xBASE+0xF0A0000	0xBASE+0xF0BFFFF	128	RESERVED	
0xBASE+0xF0C0000	0xBASE+0xF0DFFFF	128		
0xBASE+0xF0C0000	0xBASE+0xFFFFFFF	15616	RESERVED	

42.2.1 Memory Map

42.2.2 Register Summary

The conventions in Figure 42-2 and Table 42-2 serve as a key for the register summary and individual register diagrams.

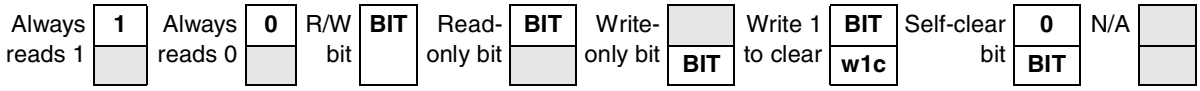


Figure 42-2. Key to Register Fields

Table 42-2 provides a key for register figures and tables and the register summary.

Table 42-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).

Table 42-2. Register Conventions (continued)

Convention	Description
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

42.2.3 Register Descriptions

42.2.3.1 IPUv3EX common registers

42.2.3.1.1 Configuration Register (IPU_CONF)

Address 0xBASE+0xE000000 (IPU_CONF) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI_SEL	IC_INPUT	CSI1_DATA_SOURCE	CSI0_DATA_SOURCE	VDI_DMFC_SYNC	IC_DMFC_SYNC	IC_DMFC_SEL	ISP_DOUBLE_FLOW	0	IDMAC_DISABLE	IPU_DIAGBUS_ON	IPU_DIAGBUS_MODE				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	VDI_EN	SISG_EN	DMFC_EN	DC_EN	SMFC_EN	DI1_EN	DI0_EN	DP_EN	ISP_EN	IRT_EN	IC_EN	CSI1_EN	CSI0_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-3. Configuration Register (IPU_CONF)

Table 42-3. Register Field Descriptions

Field	Description
31 CSI_SEL	CSI select bit; This bit selects manually between the 2 CSIs. This bit defines which CSI is the input to the IC. This bit is effective only if IC_INPUT is bit cleared 0 - CSI0 is selected 1 - CSI1 is selected
30 IC_INPUT	IC Input select bit; This bit selects manually between the 2 inputs to the IC 0 - CSI0/1 is selected; In order to select between the CSIs use the CSI_SEL bit. 1 - VDI is selected
29 CSI1_DATA_SOURCE	0
28 CSI0_DATA_SOURCE	0
27 VDI_DMFC_SYNC	This bit enables the direct path VDI -> IC_VF -> DMFC for sync flow. If this bit is set IC_DMFC_SEL must be set. 0 - the flow is disabled 1 - the flow is enabled
26 IC_DMFC_SYNC	IC to DMFC Sync flow This bit defines if the direct flow between IC to DMFC is synchronous or asynchronous 0 - async flow 1 - Sync flow
25 IC_DMFC_SEL	IC to DMFC select Selects the DMAIC_1 (channel 21) channel's connectivity between the IC and the DMFC 0 DMAIC_1 (channel 21) is routed to the IDMAC 1 DMAIC_1 (channel 21) is routed to DMFC In case DMFC was selected the IDMAC_CH_EN[21] must be clear.
24	
23	Reserved, should be cleared.
22 IDMAC_DISABLE	Image DMA controller (IDMAC) disable bit. This bit allows the user to turn off the clock of the IDMAC if the use case permits it. By default the IDMAC is enabled. 0 - IDMAC is enabled 1 - IDMAC is disabled
21 IPU_DIAGBUS_ON	IPUv3EX Diagnostics bus on This bit is connected to the IPUv3EX's output. it can be used by the SoC in order to control the iomux and bring the IPUv3EX's diagnostics bus to the SoC's pins. 1 - diagnostics bus is on 0 - diagnostics bus is off

Table 42-3. Register Field Descriptions (continued)

Field	Description
20–16 IPU_DIAGBUS_MODE	IPUv3EX diagnostic bus mode. This 5 bits select one of 16 groups of signals to be routed to the IPUv3EX's diagnostics bus 00000 - Route group 0 to the IPUv3EX's diagnostics bus 00001 - Route group 1 to the IPUv3EX's diagnostics bus ... 01111 - Route group 15 to the IPUv3EX's diagnostics bus 10000 - Route group 16 to the IPUv3EX's diagnostics bus 10001 - Route group 17 to the IPUv3EX's diagnostics bus 10010 - 11111 Reserved
15-13	Reserved, should be cleared.
12 VDI_EN	VDI enable bit.This bit must be cleared if the ISP_EN bit is set. 0 - VDI is disabled 1 - VDI is enabled
11 SISG_EN	Still Image Synchronization Generator (SISG) Enable bit 0 - SISG is disabled 1 - SISG is enabled
10 DMFC_EN	Display's Multi FIFO Controller Module (DMFC) Enable bit 0 - DMFC is disabled 1 - DMFC is enabled
9 DC_EN	Display Controller Module (DC) Enable bit 0 - DC is disabled 1 - DC is enabled
8 SMFC_EN	Sensor's Multi FIFO Controller Module (SMFC) Enable bit 0 - SMFC is disabled 1 - SMFC is enabled
7 DI1_EN	Display Interface Module 1 Enable bit 0 - DI1 is disabled 1 - DI1 is enabled
6 DI0_EN	Display interface Module 0 Enable bit 0 - DI0 is disabled 1 - DI0 is enabled
5 DP_EN	Display processor Module Enable bit 0 - DP is disabled 1 - DP is enabled
4	
3 IRT_EN	Image Rotation Module Enable bit 0 - IRT is disabled 1 - IRT is enabled
2 IC_EN	Image Conversion Module Enable bit 0 - IC is disabled 1 - IC is enabled

Table 42-3. Register Field Descriptions (continued)

Field	Description
1 CSI1_EN	Camera Sensor Interface 1 Enable bit 0 -CSI1 is disabled 1 - CSI1 is enabled
0 CSI0_EN	Camera Sensor Interface 0 Enable bit 0 -CSI0 is disabled 1 - CSI0 is enabled

42.2.3.1.2 SISG Control 0 Register (SISG_CTRL0)

Address 0xBASE+0xE000004 (SISG_CTRL0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	EXT_ACTV	MCU_ACTV_TRIG	VAL_STOP_SISG_COUNTER												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VAL_STOP_SISG_COUNTER												NO_OF_VSYNC			VSYNC_RESET_COUNTER
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-6. SISG Control 0 Register (SISG_CTRL0)

Table 42-4. Register Field Descriptions

Field	Description
30 EXT_ACTV	External Active Define if an external active trigger will start the counters. The external active trigger is an input signal to the IPUv3EX called ext_actv_trig
29 MCU_ACTV_TRIG	Reserved, should be cleared.

Table 42-4. Register Field Descriptions (continued)

Field	Description
28–4 VAL_STOP_SISG_COUNTER	SISG Stop Counters value. This is a predefined value that stops the SISG counters. The user should write to this field the N-1 value of the desired value.
3–1 NO_VSYNC_2_STRT_CNT	VSYCs to Start Counter This bits define how many VSYNCs signals will be counter before activating the SISG counters. If set to 0 starts immediately. If set to N (1..7) starts after N VSYNCs.
0 VSYNC_RST_CNT	VSYNC Resets counters Defines if the counters are stooped following VSYNC or when the counters reach a pre defined value (VAL_STOP_SISG_COUNTER) 1 The counters are stopped at VSYNC 0 The counters are stooped when the counters reach the VAL_STOP_SISG_COUNTER value.

42.2.3.1.3 SISG Control 1 Register (SISG_CTRL1)

Address 0xBASE+0xE000008 (SISG_CTRL1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SISG_OUT_POL						0	0	0	SISG_STROBE_CNT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-7. SISG Control 1 Register (SISG_CTRL1)

Table 42-5. Register Field Descriptions

Field	Description
31-5	Reserved, should be cleared.
13-8 SISG_OUT_POL	SISG_OUT_POL This bits defines the polarity of the SISG output signals 1 - active high 0 - active low
4-0 SISG_STROBE_CNT	SISG Strobe Count This fields defines the The SISG can repeat the sequence for up to 32 cycles, this is used for generating a train of pulses. This field defines the amount of cycles

42.2.3.1.4 SISG Set <i> Register (SISG_SET <i>)

Address 0xBASE+0xE00000C (SISG_SET_<i>)
offset: 4
range 1 .. 6

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	SISG_SET_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SISG_SET_<i>															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-8. SISG Set <i> Register (SISG_SET <i>)

Table 42-6. Register Field Descriptions

Field	Description
31–24	Reserved, should be cleared.
23–0 SISG_SET<i>	SISG SET <i> value These bits define the set value of the SISG counter #<i>

42.2.3.1.5 SISG Clear <i> Register (SISG_CLR <i>)

Address 0xBASE+0xE000024 (SISG_CLR_<i>)
offset: 4
range 1 .. 6

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	SISG_CLEAR_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SISG_CLEAR_<i>															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-9. SISG Clear <i> Register (SISG_CLR <i>)

Table 42-7. Register Field Descriptions

Field	Description
31–24	Reserved, should be cleared.
23–0 SISG_CLEAR<i>	SISG CLR <i> value These bits define the clear value of the SISG counter #<i>

42.2.3.1.6 Interrupt Control Register 1 (IPU_INT_CTRL_1)

This register contains part of IPUv3EX interrupts controls. The controls of EOF (end of frame) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE00003C (IPU_INT_CTRL_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOF_EN_31	0	IDMAC_EOF_EN_29	IDMAC_EOF_EN_28	IDMAC_EOF_EN_27	0	0	IDMAC_EOF_EN_24	IDMAC_EOF_EN_23	IDMAC_EOF_EN_22	IDMAC_EOF_EN_21	IDMAC_EOF_EN_20	0	IDMAC_EOF_EN_18	IDMAC_EOF_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOF_EN_15	IDMAC_EOF_EN_14	IDMAC_EOF_EN_13	IDMAC_EOF_EN_12	IDMAC_EOF_EN_11	IDMAC_EOF_EN_10	IDMAC_EOF_EN_9	IDMAC_EOF_EN_8	IDMAC_EOF_EN_7	IDMAC_EOF_EN_6	IDMAC_EOF_EN_5	IDMAC_EOF_EN_4	IDMAC_EOF_EN_3	IDMAC_EOF_EN_2	IDMAC_EOF_EN_1	IDMAC_EOF_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-10. Interrupt Control Register 1 (IPU_INT_CTRL_1)

Figure 42-11.

Table 42-8. IPU_INT_CTRL_1 Field Descriptions

Field	Description
31–0 *_EOF_EN[1]	Enable End of Frame of Channel interrupt. This bit is the control of End Of Frame of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

1 Indicates the corresponding DMA channel number.

42.2.3.1.7 Interrupt Control Register 2 (IPU_INT_CTRL_2)

This register contains part of IPUv3EX interrupts controls. The controls of EOF (end of frame) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000040 (IPU_INT_CTRL_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOF_EN_52	IDMAC_EOF_EN_51	IDMAC_EOF_EN_50	IDMAC_EOF_EN_49	IDMAC_EOF_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOF_EN_47	IDMAC_EOF_EN_46	IDMAC_EOF_EN_45	IDMAC_EOF_EN_44	IDMAC_EOF_EN_43	IDMAC_EOF_EN_42	IDMAC_EOF_EN_41	IDMAC_EOF_EN_40	0	0	0	0	0	0	IDMAC_EOF_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-12. Interrupt Control Register 2 (IPU_INT_CTRL_2)

Table 42-9. IPU_INT_CTRL_2 Field Descriptions

Field	Description
31–0 *_EOF_EN[1]	Enable End of Frame of Channel interrupt. This bit is the control of End Of Frame of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

1 Indicates the corresponding DMA channel number.

42.2.3.1.8 Interrupt Control Register 3 (IPU_INT_CTRL_3)

This register contains part of IPUv3EX interrupts controls. The controls of NFACK (New Frame Ack) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000044 (IPU_INT_CTRL_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_NFACK_EN_31	0	IDMAC_NFACK_EN_29	IDMAC_NFACK_EN_28	IDMAC_NFACK_EN_27	0	0	IDMAC_NFACK_EN_24	IDMAC_NFACK_EN_23	IDMAC_NFACK_EN_22	IDMAC_NFACK_EN_21	IDMAC_NFACK_EN_20	0	IDMAC_NFACK_EN_18	IDMAC_NFACK_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFACK_EN_15	IDMAC_NFACK_EN_14	IDMAC_NFACK_EN_13	IDMAC_NFACK_EN_12	IDMAC_NFACK_EN_11	IDMAC_NFACK_EN_10	IDMAC_NFACK_EN_9	IDMAC_NFACK_EN_8	IDMAC_NFACK_EN_7	IDMAC_NFACK_EN_6	IDMAC_NFACK_EN_5	IDMAC_NFACK_EN_4	IDMAC_NFACK_EN_3	IDMAC_NFACK_EN_2	IDMAC_NFACK_EN_1	IDMAC_NFACK_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-13. Interrupt Control Register 3 (IPU_INT_CTRL_3)

Table 42-10. IPU_INT_CTRL_3 Field Descriptions

Field	Description
31–0 *_NFACK_EN[1]	Enable New Frame Ack of Channel interrupt. This bit is the control of New Frame Ack of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.9 Interrupt Control Register 4 (IPU_INT_CTRL_4)

This register contains part of IPUv3EX interrupts controls. The controls of NFACK (New Frame Ack) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000048 (IPU_INT_CTRL_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_NFACK_EN_52	IDMAC_NFACK_EN_51	IDMAC_NFACK_EN_50	IDMAC_NFACK_EN_49	IDMAC_NFACK_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFACK_EN_47	IDMAC_NFACK_EN_46	IDMAC_NFACK_EN_45	IDMAC_NFACK_EN_44	IDMAC_NFACK_EN_43	IDMAC_NFACK_EN_42	IDMAC_NFACK_EN_41	IDMAC_NFACK_EN_40	0	0	0	0	0	0	IDMAC_NFACK_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-14. Interrupt Control Register 4 (IPU_INT_CTRL_4)

Table 42-11. IPU_INT_CTRL_4 Field Descriptions

Field	Description
31–0 *_NFACK_EN[1]	Enable New Frame Ack of Channel interrupt. This bit is the control of New Frame Ack of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

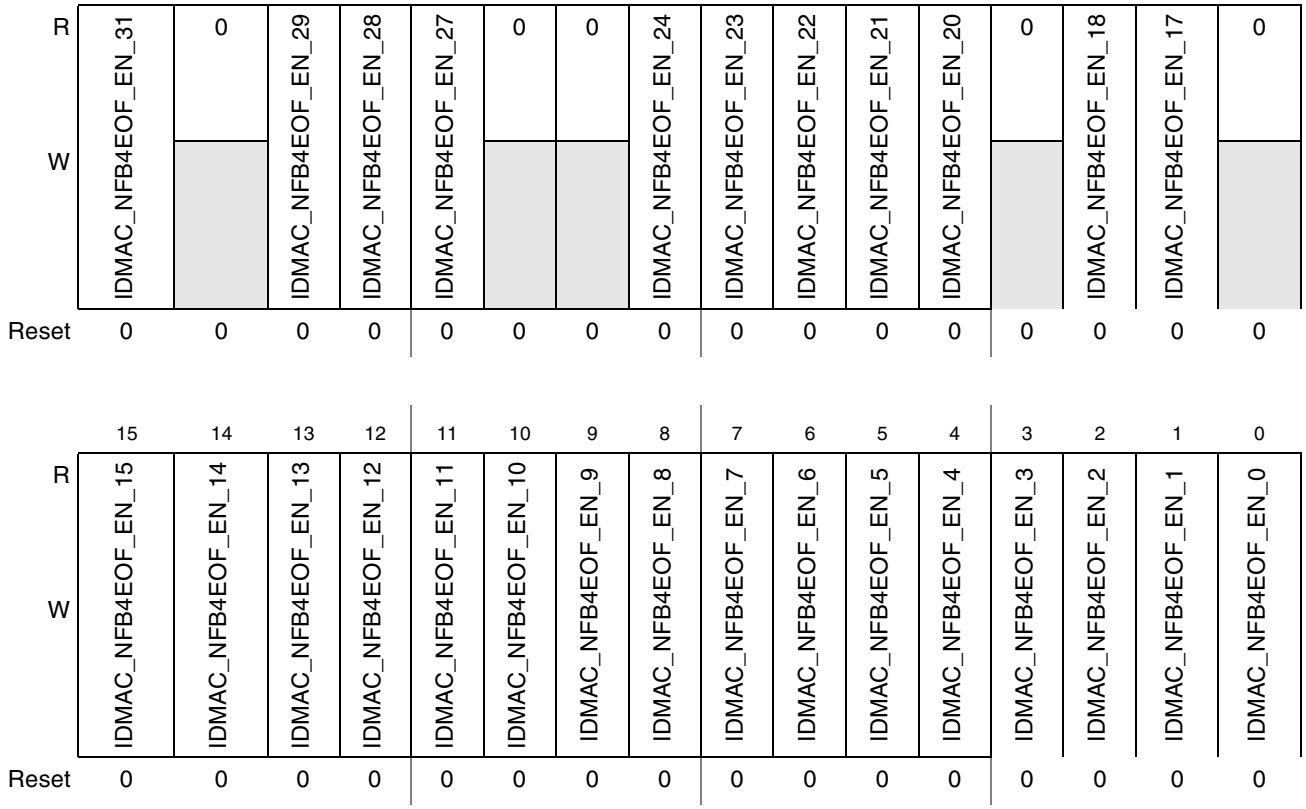
[1] Indicates the corresponding DMA channel number.

42.2.3.1.10 Interrupt Control Register 5 (IPU_INT_CTRL_5)

This register contains part of IPUv3EX interrupts controls. The controls of the New-frame before end-of-frame indication (NFB4EOF) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE00004C (IPU_INT_CTRL_5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16



	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_NFB4EOF_EN_31	0	IDMAC_NFB4EOF_EN_29	IDMAC_NFB4EOF_EN_28	IDMAC_NFB4EOF_EN_27	0	0	IDMAC_NFB4EOF_EN_24	IDMAC_NFB4EOF_EN_23	IDMAC_NFB4EOF_EN_22	IDMAC_NFB4EOF_EN_21	IDMAC_NFB4EOF_EN_20	0	IDMAC_NFB4EOF_EN_18	IDMAC_NFB4EOF_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFB4EOF_EN_15	IDMAC_NFB4EOF_EN_14	IDMAC_NFB4EOF_EN_13	IDMAC_NFB4EOF_EN_12	IDMAC_NFB4EOF_EN_11	IDMAC_NFB4EOF_EN_10	IDMAC_NFB4EOF_EN_9	IDMAC_NFB4EOF_EN_8	IDMAC_NFB4EOF_EN_7	IDMAC_NFB4EOF_EN_6	IDMAC_NFB4EOF_EN_5	IDMAC_NFB4EOF_EN_4	IDMAC_NFB4EOF_EN_3	IDMAC_NFB4EOF_EN_2	IDMAC_NFB4EOF_EN_1	IDMAC_NFB4EOF_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-15. Interrupt Control Register 5 (IPU_INT_CTRL_5)

Table 42-12. IPU_INT_CTRL_5 Field Descriptions

Field	Description
31–0 *_NFB4EOF_EN[1]	New Frame before end-of-frame error indication of Channel interrupt. This bit is the control of New Frame before end-of-frame error interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.11 Interrupt Control Register 6 (IPU_INT_CTRL_6)

This register contains part of IPUv3EX interrupts controls. The controls of the New-frame before end-of-frame indication (NFB4EOF_ERR) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000050 (IPU_INT_CTRL_6) Access: User read-write

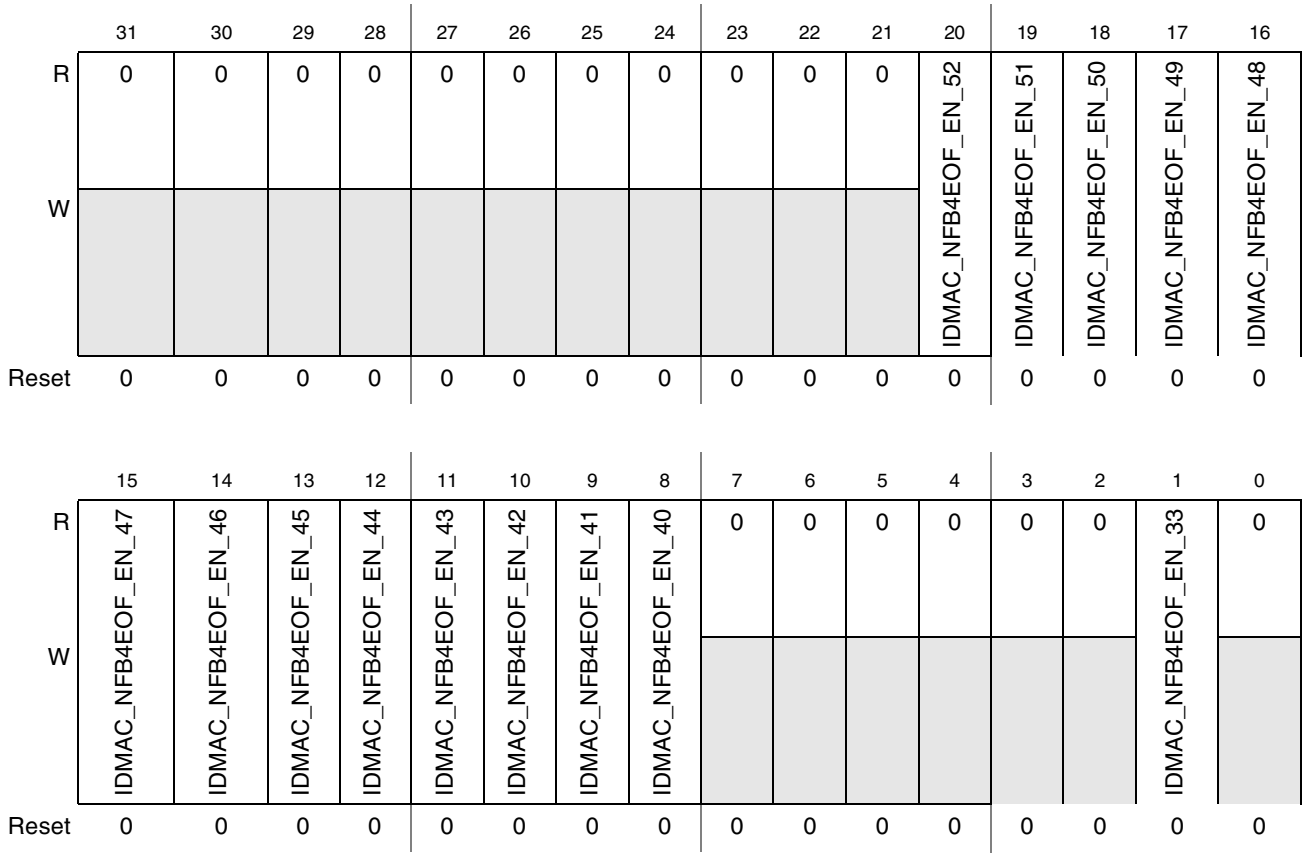


Figure 42-16. Interrupt Control Register 6 (IPU_INT_CTRL_6)

Table 42-13. IPU_INT_CTRL_6 Field Descriptions

Field	Description
31–0 *_NFB4EOF_EN[1]	New Frame before end-of-frame error indication of Channel interrupt. This bit is the control of New Frame before end-of-frame error interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.12 Interrupt Control Register 7(IPU_INT_CTRL_7)

This register contains part of IPUv3EX interrupts controls. The controls of the End-of-Scroll indication (EOS) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000054 (IPU_INT_CTRL_7) Access: User read-write

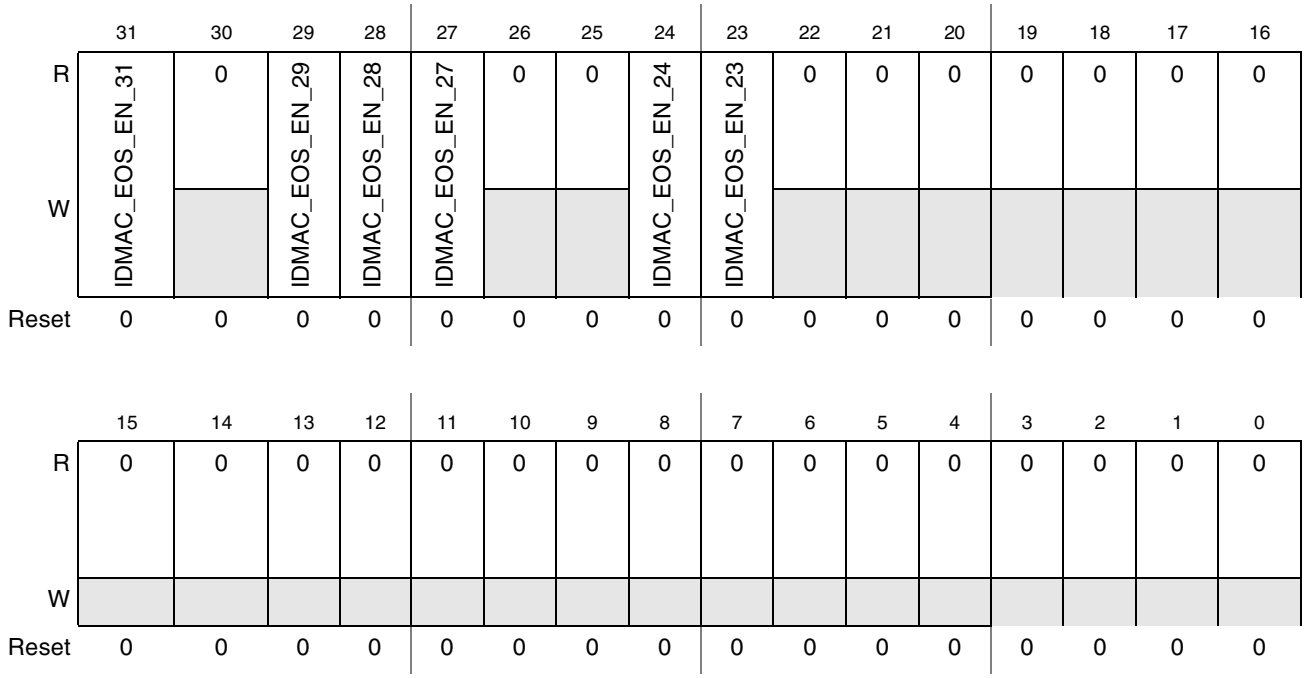


Figure 42-17. Interrupt Control Register 7(IPU_INT_CTRL_7)

Table 42-14. IPU_INT_CTRL_7 Field Descriptions

Field	Description
31–0 *_EOS_EN[1]	End of Scroll indication of Channel interrupt. This bit is the control of End of Scroll interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.13 Interrupt Control Register 8 (IPU_INT_CTRL_8)

This register contains part of IPUv3EX interrupts controls. The controls of the End of Scroll indication (EOS) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000058 (IPU_INT_CTRL_8) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOS_EN_52	IDMAC_EOS_EN_51	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOS_EN_44	IDMAC_EOS_EN_43	IDMAC_EOS_EN_42	IDMAC_EOS_EN_41	0	0	0	0	0	0	0	IDMAC_EOS_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-18. Interrupt Control Register 8 (IPU_INT_CTRL_8)

Table 42-15. IPU_INT_CTRL_8 Field Descriptions

Field	Description
31–0 *_EOS_EN[1]	End of Scroll of Channel interrupt. This bit is the control of End Of Scroll interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.14 Interrupt Control Register 9 (IPU_INT_CTRL_9)

This register contains part of IPUv3EX interrupts controls. This register controls error interrupt signals coming from different modules within IPUv3EX.

Address 0xBASE+0xE00005C (IPU_INT_CTRL_9) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI1_PUPE_EN	CSI0_PUPE_EN	ISP_PUPE_EN	IC_VF_BUF_OVF_EN	IC_ENC_BUF_OVF_EN	IC_BAYER_BUF_OVF_EN	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VDI_FIFO1_OVF_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-19. Interrupt Control Register 9 (IPU_INT_CTRL_9)

Table 42-16. IPU_INT_CTRL_9 Field Descriptions

Field	Description
31 CSI1_PUPE_EN	CSI1_PUPE_EN - CSI1 parameters update error interrupt enable. This bit enables an interrupt that is a result of an error generated by the CSI1. The error is generated in case where new frame arrived from the CSI1 before the completion of the CSI1's parameters update by the SRM 0 Interrupt is disabled. 1 Interrupt is enabled.
30 CSI0_PUPE_EN	CSI0_PUPE_EN - CSI0 parameters update error interrupt enable. This bit enables an interrupt that is a result of an error generated by the CSI0. The error is generated in case where new frame arrived from the CSI0 before the completion of the CSI0's parameters update by the SRM 0 Interrupt is disabled. 1 Interrupt is enabled.
29	
28 IC_VF_BUF_OVF_EN	This bit enables an interrupt that is a result of the IC Buffer overflow for view finder coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is disabled. 1 Interrupt is enabled.

Table 42-16. IPU_INT_CTRL_9 Field Descriptions (continued)

Field	Description
27 IC_ENC_BUF_OVF_EN	This bit enables an interrupt that is a result of the IC Buffer overflow for encoding coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is disabled. 1 Interrupt is enabled.
26 IC_BAYER_BUF_OVF_EN	This bit enables an interrupt that is a result of the IC Buffer overflow for bayer coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is disabled. 1 Interrupt is enabled.
25-1	Reserved
0 VDI_FIFO1_OVF_EN	FIFO1 overflow Interrupt1 Enable The VDI generates FIFO1 overflow interrupt1 when the write pointer of FIFO1 overruns read pointer. 0 Interrupt is disabled. 1 Interrupt is enabled.

42.2.3.1.15 Interrupt Control Register 10 (IPU_INT_CTRL_10)

This register contains part of IPUv3EX interrupts controls. This register controls error interrupt signals coming from different modules within IPUv3EX.

Address 0xBASE+0xE000060 (IPU_INT_CTRL_10) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	ISP_RAM_HIST_OF_EN	ISP_RAM_ST_OF_EN	SMFC3_FRM_LOST_EN	SMFC2_FRM_LOST_EN	SMFC1_FRM_LOST_EN	SMFC0_FRM_LOST_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

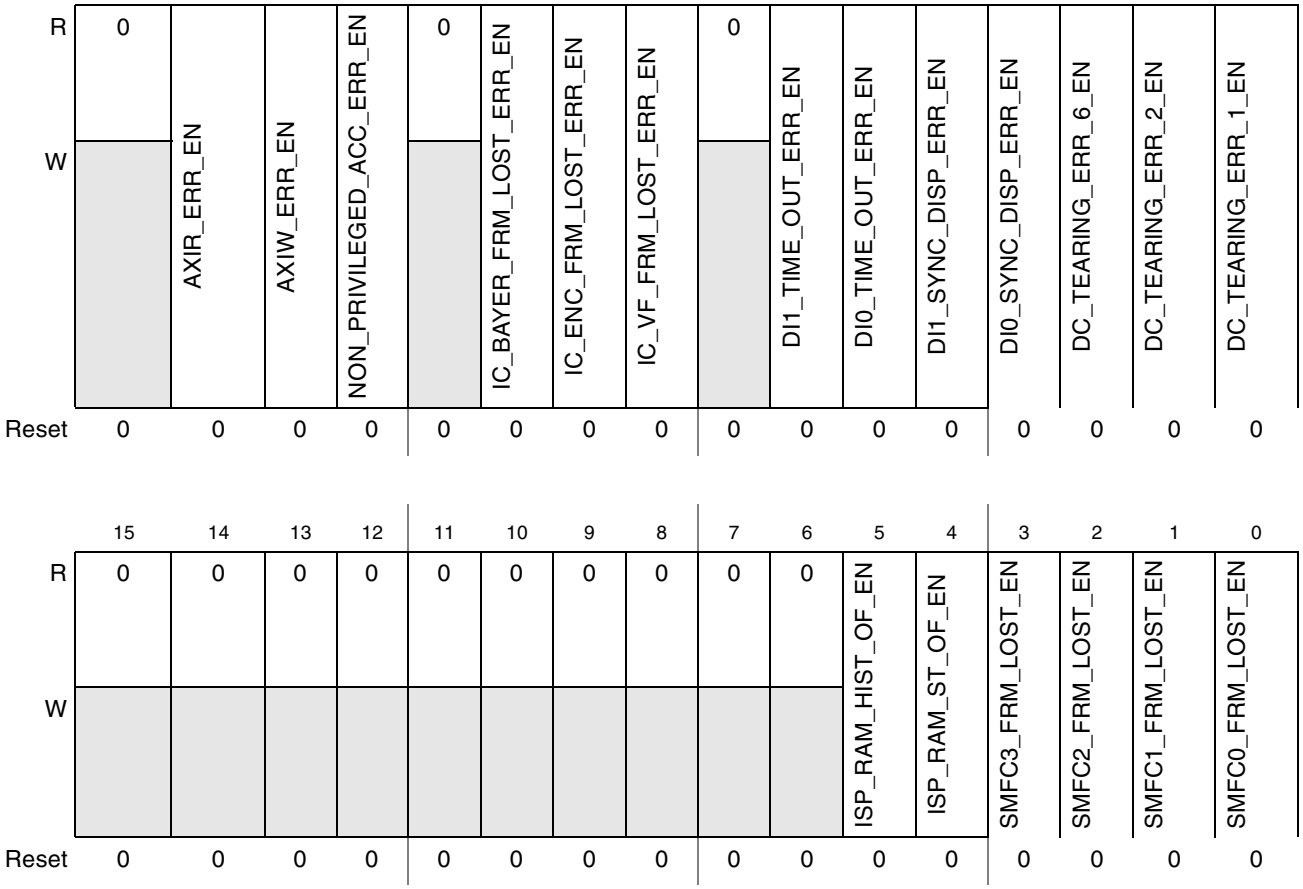


Figure 42-20. Interrupt Control Register 10 (IPU_INT_CTRL_10)

Table 42-17. IPU_INT_CTRL_10 Field Descriptions

Field	Description
31	Reserved
30 AXIR_ERR_EN	This bit enables an interrupt that is a result of AXI read access resulted with error response. 0 Interrupt is disabled. 1 Interrupt is enabled.
29 AXIW_ERR_EN	This bit enables an interrupt that is a result of AXI write access resulted with error response. 0 Interrupt is disabled. 1 Interrupt is enabled.
28 NON_PRIVILEGED_ACC_ERR_EN	Non Privileged Access Error interrupt enable. The CPMEM and the DP can be accessed by the CPU in privileged mode only HPROT[1] =1. An attempt to access these regions in user mode will issue an interrupt. This bit enables the interrupt. 0 Interrupt is disabled. 1 Interrupt is enabled
27	Reserved

Table 42-17. IPU_INT_CTRL_10 Field Descriptions (continued)

Field	Description
26 IC_BAYER_FRM_LOST_ERR_EN	This bit enables an interrupt that is a result of IC's Bayer frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
25 IC_ENC_FRM_LOST_ERR_EN	This bit enables an interrupt that is a result of IC's encoding frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
24 IC_VF_FRM_LOST_ERR_EN	This bit enables an interrupt that is a result of IC's view finder frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
23	Reserved
22 DI1_TIME_OUT_ERR_EN	DI1 time out error interrupt enable This bit enables the interrupt that is a result of a time out error during a read access via DI1 0 Interrupt is disabled. 1 Interrupt is enabled.
21 DI0_TIME_OUT_ERR_EN	DI0 time outwore interrupt enable This bit enables the interrupt that is a result of a time out error during a read access via DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
20 DI1_SYNC_DISP_ERR_EN	DI1 Synchronous display error enable This bit enables the interrupt that is a result of an error during access to a synchronous display via DI1 0 Interrupt is disabled. 1 Interrupt is enabled.
19 DI0_SYNC_DISP_ERR_EN	DI0 Synchronous display error enable This bit enables the interrupt that is a result of an error during access to a synchronous display via DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
18 DC_TEARING_ERR_6_EN	Tearing Error #6 enable This bit enables the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 6 0 Interrupt is disabled. 1 Interrupt is enabled.
17 DC_TEARING_ERR_2_EN	Tearing Error #2 enable This bit enables the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 2 0 Interrupt is disabled. 1 Interrupt is enabled.
16 DC_TEARING_ERR_1_EN	Tearing Error #1 enable This bit enables the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 1 0 Interrupt is disabled. 1 Interrupt is enabled.
15–6	Reserved

Table 42-17. IPU_INT_CTRL_10 Field Descriptions (continued)

Field	Description
5 ISP_RAM_HIST_OF_EN	—
4 ISP_RAM_ST_OF_EN	—
3 SMFC3_FRM_LOST_EN	Frame Lost of SMFC channel 3 interrupt enable bit This bit enables an interrupt that is a result of a Frame Lost of SMFC channel 3. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 SMFC2_FRM_LOST_EN	Frame Lost of SMFC channel 2 interrupt enable bit This bit enables an interrupt that is a result of a Frame Lost of SMFC channel 2. 0 Interrupt is disabled. 1 Interrupt is enabled.
1 SMFC1_FRM_LOST_EN	Frame Lost of SMFC channel 1 interrupt enable bit This bit enables an interrupt that is a result of a Frame Lost of SMFC channel 1. 0 Interrupt is disabled. 1 Interrupt is enabled.
0 SMFC0_FRM_LOST_EN	Frame Lost of SMFC channel 0 interrupt enable bit This bit enables an interrupt that is a result of a Frame Lost of SMFC channel 0. 0 Interrupt is disabled. 1 Interrupt is enabled.

42.2.3.1.16 Interrupt Control Register 11 (IPU_INT_CTRL_11)

This register contains part of IPUv3EX interrupts controls. The controls of the end-of-band indication (EOBND) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000064 (IPU_INT_CTRL_11) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	IDMAC_EOBND_EN_22	IDMAC_EOBND_EN_21	IDMAC_EOBND_EN_20	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOBND_EN_12	IDMAC_EOBND_EN_11	0	0	0	0	0	IDMAC_EOBND_EN_5	0	IDMAC_EOBND_EN_3	IDMAC_EOBND_EN_2	IDMAC_EOBND_EN_1	IDMAC_EOBND_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-21. Interrupt Control Register 11 (IPU_INT_CTRL_11)

Table 42-18. IPU_INT_CTRL_11 Field Descriptions

Field	Description
31–0 *_EOBND_EN[1]	end-of-band indication of Channel interrupt. This bit is the control of end-of-band interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.17 Interrupt Control Register 12 (IPU_INT_CTRL_12)

This register contains part of IPUv3EX interrupts controls. The controls of the end-of-band indication (EOBND) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000068 (IPU_INT_CTRL_12) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOBND_EN_50	IDMAC_EOBND_EN_49	IDMAC_EOBND_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOBND_EN_47	IDMAC_EOBND_EN_46	IDMAC_EOBND_EN_45	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-22. Interrupt Control Register 12 (IPU_INT_CTRL_12)

Table 42-19. IPU_INT_CTRL_12 Field Descriptions

Field	Description
31–0 *_EOBND_EN[1]	end-of-band indication of Channel interrupt. This bit is the control end-of-band interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

1 Indicates the corresponding DMA channel number.

42.2.3.1.18 Interrupt Control Register 13 (IPU_INT_CTRL_13)

This register contains part of IPUv3EX interrupts controls. The controls of the threshold crossing indication (TH) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE00006C (IPU_INT_CTRL_13) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

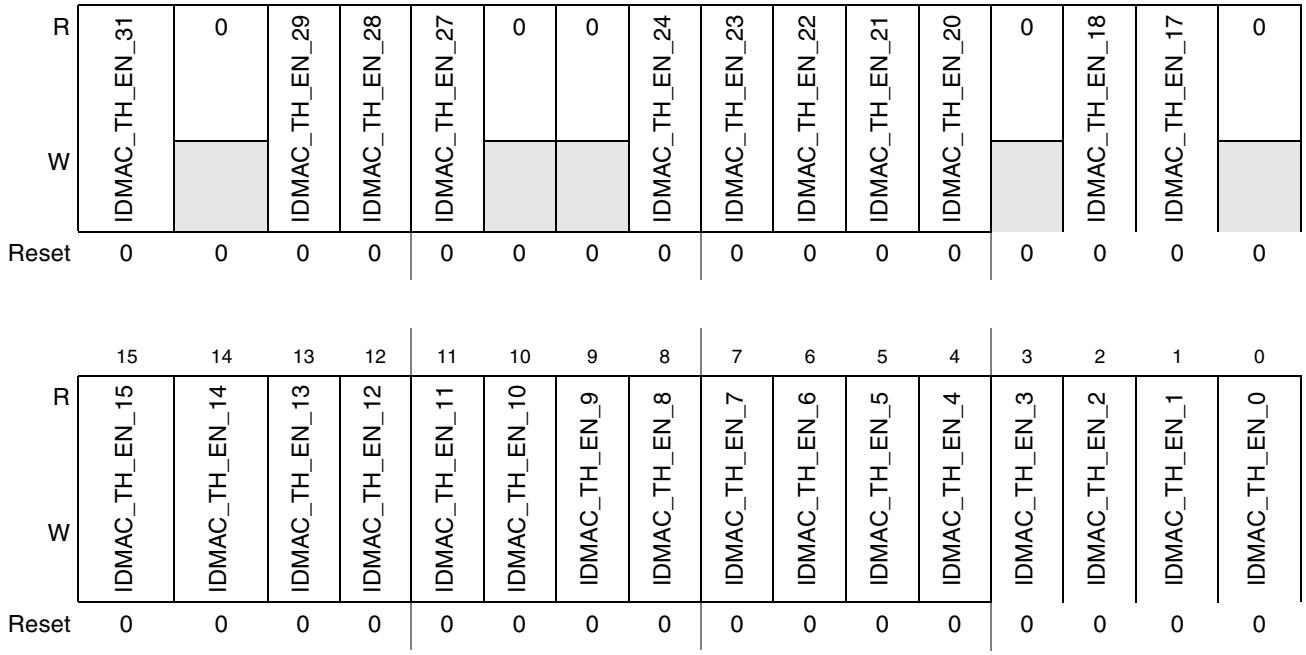


Figure 42-23. Interrupt Control Register 13 (IPU_INT_CTRL_13)

Table 42-20. IPU_INT_CTRL_13 Field Descriptions

Field	Description
31–0 *_TH_EN[1]	Threshold crossing indication of Channel interrupt. This bit is the control of Threshold crossing interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.19 Interrupt Control Register 14 (IPU_INT_CTRL_14)

This register contains part of IPUv3EX interrupts controls. The controls of the Threshold crossing indication (TH) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000070 (IPU_INT_CTRL_14) Access: User read-write

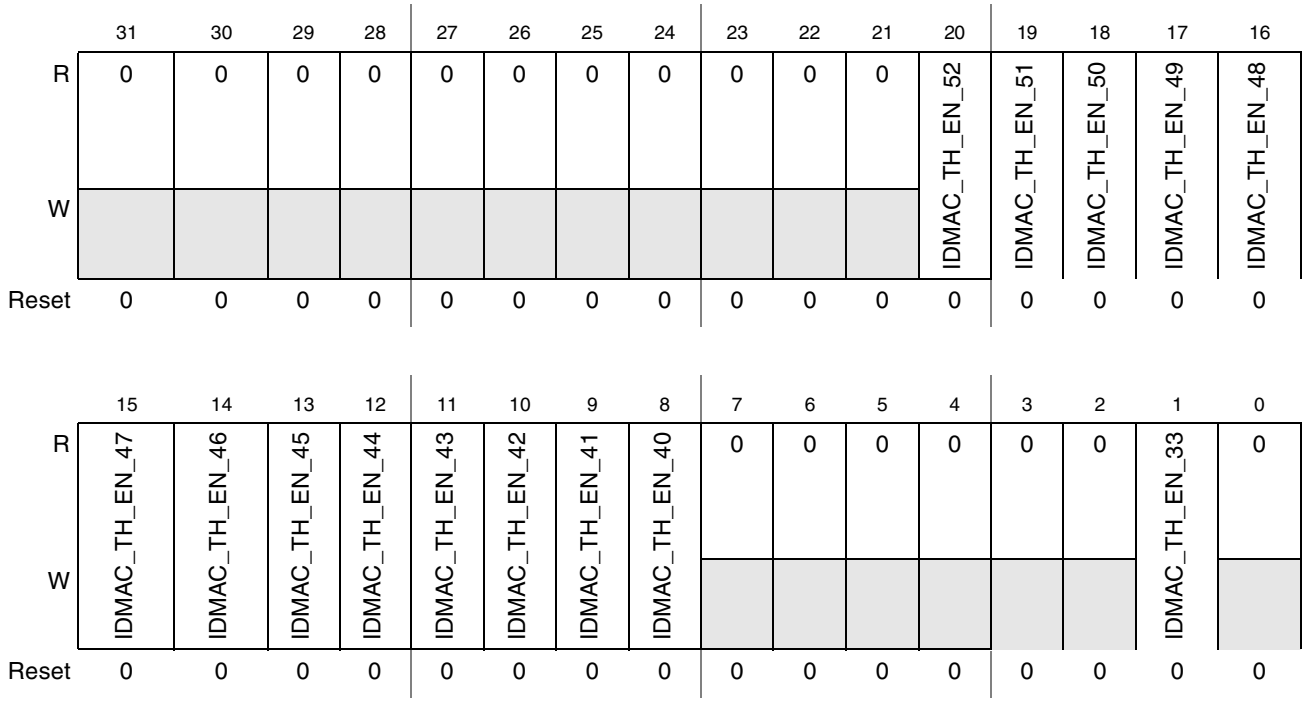


Figure 42-24. Interrupt Control Register 14 (IPU_INT_CTRL_14)

Table 42-21. IPU_INT_CTRL_14 Field Descriptions

Field	Description
31–0 *_TH_EN[1]	Threshold crossing indication of Channel interrupt. This bit is the control Threshold crossing interrupt of Channel #n. 0 Interrupt is disabled. 1 Interrupt is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.20 Interrupt Control Register 15 (IPU_INT_CTRL_15)

This register contains part of IPUv3EX interrupts controls. The controls of general purpose interrupts can be found in this register.

Address 0xBASE+0xE000074 (IPU_INT_CTRL_15) Access: User read-write

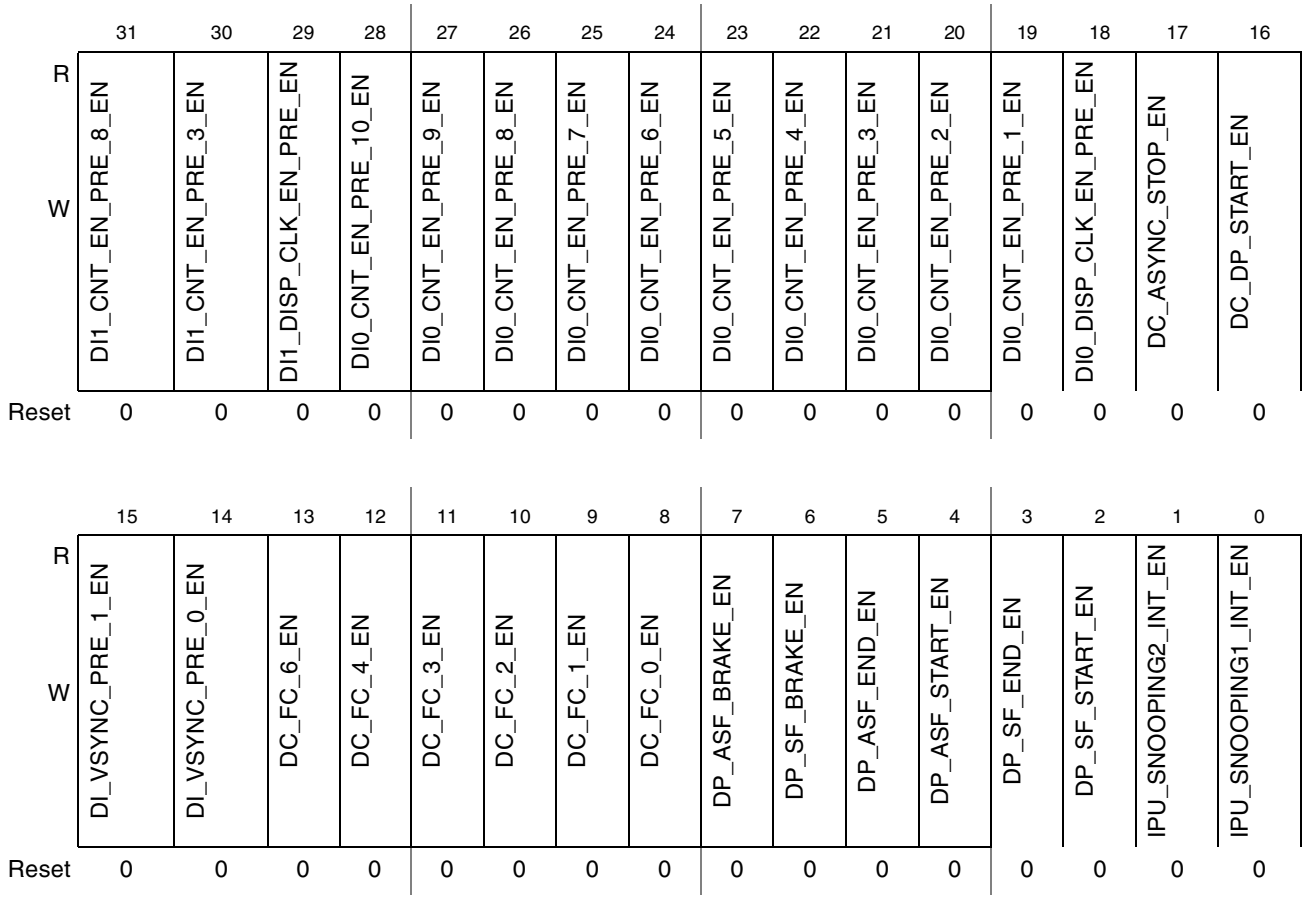


Figure 42-25. Interrupt Control Register 15 (IPU_INT_CTRL_15)

Table 42-22. IPU_INT_CTRL_15 Field Descriptions

Field	Description
31 DI1_CNT_EN_PRE_8_EN	This bit enables the interrupt that is a result of a trigger generated by counter #8 of DI1 0 Interrupt is disabled. 1 Interrupt is enabled.
30 DI1_CNT_EN_PRE_3_EN	This bit enables the interrupt that is a result of a trigger generated by counter #3 of DI1 0 Interrupt is disabled. 1 Interrupt is enabled.
29 DI1_DISP_CLK_EN_PRE_EN	0 Interrupt is disabled. 1 Interrupt is enabled.

Table 42-22. IPU_INT_CTRL_15 Field Descriptions (continued)

Field	Description
28 DI0_CNT_EN_PRE_10_EN	This bit enables the interrupt that is a result of a trigger generated by counter #10 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
27 DI0_CNT_EN_PRE_9_EN	This bit enables the interrupt that is a result of a trigger generated by counter #9 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
26 DI0_CNT_EN_PRE_8_EN	This bit enables the interrupt that is a result of a trigger generated by counter #8 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
25 DI0_CNT_EN_PRE_7_EN	This bit enables the interrupt that is a result of a trigger generated by counter #7 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
24 DI0_CNT_EN_PRE_6_EN	This bit enables the interrupt that is a result of a trigger generated by counter #6 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
23 DI0_CNT_EN_PRE_5_EN	This bit enables the interrupt that is a result of a trigger generated by counter #5 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
22 DI0_CNT_EN_PRE_4_EN	This bit enables the interrupt that is a result of a trigger generated by counter #4 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
21 DI0_CNT_EN_PRE_3_EN	This bit enables the interrupt that is a result of a trigger generated by counter #3 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
20 DI0_CNT_EN_PRE_2_EN	This bit enables the interrupt that is a result of a trigger generated by counter #2 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
19 DI0_CNT_EN_PRE_1_EN	This bit enables the interrupt that is a result of a trigger generated by counter #1 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.
18 DI0_CNT_EN_PRE_0_EN	This bit enables the interrupt that is a result of a trigger generated by counter #0 of DI0 0 Interrupt is disabled. 1 Interrupt is enabled.

Table 42-22. IPU_INT_CTRL_15 Field Descriptions (continued)

Field	Description
17 DC_ASYNC_STOP_EN	This bit enables the interrupt asserted anytime the DP stops an async flow and moves to a sync flow 0 Interrupt is disabled. 1 Interrupt is enabled.
16 DC_DP_START_EN	This bit enables the interrupt asserted anytime the DP start a new sync or async flow or when an async flow is interrupted by a sync flow 0 Interrupt is disabled. 1 Interrupt is enabled.
15 DI_VSYNC_PRE_1_EN	This bit enables the DI1 interrupt indicating of a VSYNC signal asserted 2 rows before the VSYNC sent to the display 0 Interrupt is disabled. 1 Interrupt is enabled.
14 DI_VSYNC_PRE_0_EN	This bit enables the DI0 interrupt indicating of a VSYNC signal asserted 2 rows before the VSYNC sent to the display 0 Interrupt is disabled. 1 Interrupt is enabled.
13 DC_FC_6_EN	This bit enables the DC Frame Complete on channel #6 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
12 DC_FC_4_EN	This bit enables the DC Frame Complete on channel #4 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
11 DC_FC_3_EN	This bit enables the DC Frame Complete on channel #3 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
10 DC_FC_2_EN	This bit enables the DC Frame Complete on channel #2 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
9 DC_FC_1_EN	This bit enables they'd Frame Complete on channel #1 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
8 DC_FC_0_EN	This bit enables they'd Frame Complete on channel #0 interrupt 0 Interrupt is disabled. 1 Interrupt is enabled.
7 DP_ASF_BRAKE_EN	DP Async Flow Brake enable bit. This bit enables the interrupt that is a result of the async flow brake at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.
6 DP_SF_BRAKE_EN	DP Sync Flow Brake enable bit. This bit enables the interrupt that is a result of the Sync flow brake at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.
5 DP_ASF_END_EN	DP Async Flow End enable bit. This bit enables the interrupt that is a result of the Async flow end at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.

Table 42-22. IPU_INT_CTRL_15 Field Descriptions (continued)

Field	Description
4 DP_ASF_START_EN	DP Async Flow Start enable bit. This bit enables the interrupt that is a result of the Async flow start at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.
3 DP_SF_END_EN	DP Sync Flow End enable bit. This bit enables the interrupt that is a result of the Sync flow end at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.
2 DP_SF_START_EN	DP Sync Flow Start enable bit. This bit enables the interrupt that is a result of the Sync flow start at the DP 0 Interrupt is disabled. 1 Interrupt is enabled.
1 IPU_SNOOPING2_INT_EN	IPUv3EX snooping 2 interrupt enable bit. This bit enables the interrupt that is a result of the detection of a snooping 2 signal assertion coming to the IPUv3EX 0 Interrupt is disabled. 1 Interrupt is enabled.
0 IPU_SNOOPING1_INT_EN	IPUv3EX snooping 1 interrupt enable bit. This bit enables the interrupt that is a result of the detection of a snooping 1 signal assertion coming to the IPUv3EX 0 Interrupt is disabled. 1 Interrupt is enabled.

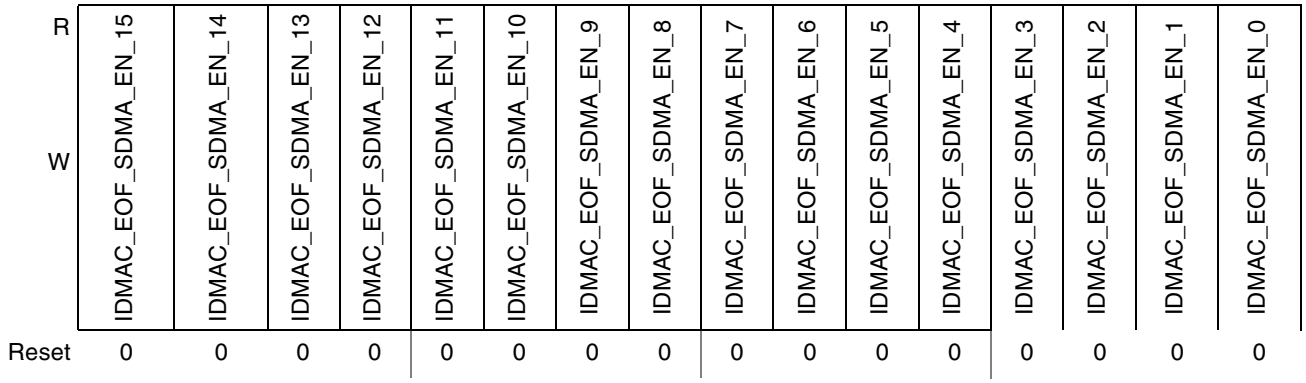
42.2.3.1.21 SDMA Event Control Register 1 (IPU_SDMA_EVENT_1)

This register contains part of IPUv3EX SDMA events controls. The controls of EOF (end of frame) of DMA Channels SDMA events[31:0] can be found in this register.

Address 0xBASE+0xE000078 (IPU_SDMA_EVENT_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOF_SDMA_EN_31	0	IDMAC_EOF_SDMA_EN_29	IDMAC_EOF_SDMA_EN_28	IDMAC_EOF_SDMA_EN_27	0	0	IDMAC_EOF_SDMA_EN_24	IDMAC_EOF_SDMA_EN_23	IDMAC_EOF_SDMA_EN_22	IDMAC_EOF_SDMA_EN_21	IDMAC_EOF_SDMA_EN_20	0	IDMAC_EOF_SDMA_EN_18	IDMAC_EOF_SDMA_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0



	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOF_SDMA_EN_15	IDMAC_EOF_SDMA_EN_14	IDMAC_EOF_SDMA_EN_13	IDMAC_EOF_SDMA_EN_12	IDMAC_EOF_SDMA_EN_11	IDMAC_EOF_SDMA_EN_10	IDMAC_EOF_SDMA_EN_9	IDMAC_EOF_SDMA_EN_8	IDMAC_EOF_SDMA_EN_7	IDMAC_EOF_SDMA_EN_6	IDMAC_EOF_SDMA_EN_5	IDMAC_EOF_SDMA_EN_4	IDMAC_EOF_SDMA_EN_3	IDMAC_EOF_SDMA_EN_2	IDMAC_EOF_SDMA_EN_1	IDMAC_EOF_SDMA_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-26. SDMA Event Control Register 1 (IPU_SDMA_EVENT_1)

Table 42-23. IPU_SDMA_EVENT_1 Field Descriptions

Field	Description
31–0 *_EOF_SDMA_EN[1]	Enable End of Frame of Channel SDMA event. This bit is the control of End Of Frame of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.22 SDMA Event Control Register 2 (IPU_SDMA_EVENT_2)

This register contains part of IPUv3EX SDMA events controls. The controls of EOF (end of frame) of DMA Channels SDMA events [63:32] can be found in this register.

Address 0xBASE+0xE00007C (IPU_SDMA_EVENT_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOF_SDMA_EN_52	IDMAC_EOF_SDMA_EN_51	IDMAC_EOF_SDMA_EN_50	IDMAC_EOF_SDMA_EN_49	IDMAC_EOF_SDMA_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOF_SDMA_EN_47	IDMAC_EOF_SDMA_EN_46	IDMAC_EOF_SDMA_EN_45	IDMAC_EOF_SDMA_EN_44	IDMAC_EOF_SDMA_EN_43	IDMAC_EOF_SDMA_EN_42	IDMAC_EOF_SDMA_EN_41	IDMAC_EOF_SDMA_EN_40	0	0	0	0	0	0	IDMAC_EOF_SDMA_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-27. SDMA Event Control Register 2 (IPU_SDMA_EVENT_2)

Table 42-24. IPU_SDMA_EVENT_2 Field Descriptions

Field	Description
31–0 *_EOF_SDMA_EN[1]	Enable End of Frame of Channel SDMA event. This bit is the control of End Of Frame of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.23 SDMA Event Control Register 3 (IPU_SDMA_EVENT_3)

This register contains part of IPUv3EX SDMA events controls. The controls of NFACK (New Frame Acknowledge) of DMA Channels SDMA events[31:0] can be found in this register.

Address 0xBASE+0xE000080 (IPU_SDMA_EVENT_3) Access: User read-write

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

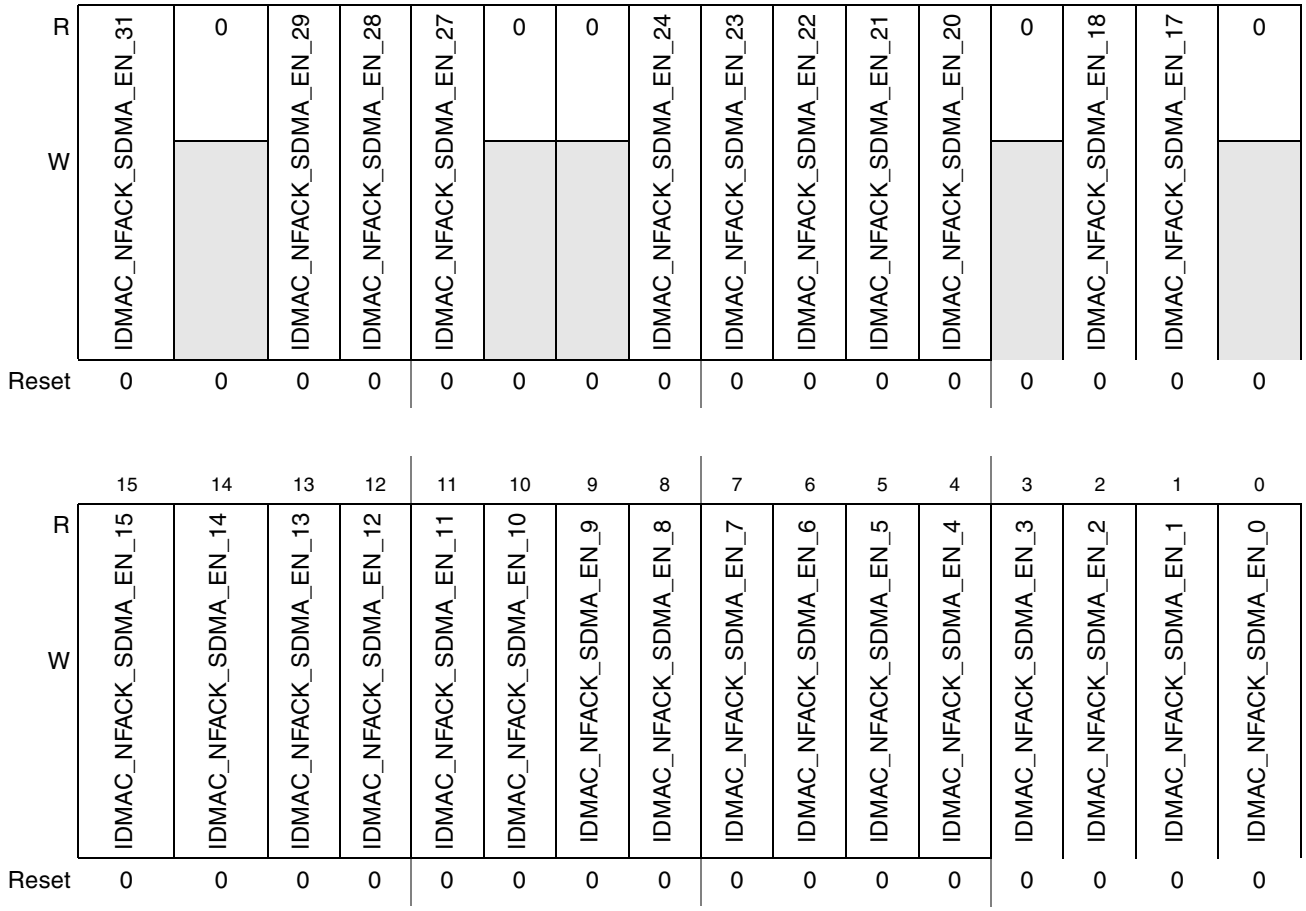


Figure 42-28. SDMA Event Control Register 3 (IPU_SDMA_EVENT_3)

Table 42-25. IPU_SDMA_EVENT_3 Field Descriptions

Field	Description
31–0 *_NFACK_SDMA_EN[1]	Enable New Frame Acknowledge of Channel SDMA event. This bit is the control of New Frame Acknowledge of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.24 SDMA Event Control Register 4 (IPU_SDMA_EVENT_4)

This register contains part of IPUv3EX SDMA events controls. The controls of NFACK (New Frame Acknowledge) of DMA Channels SDMA events [63:32] can be found in this register.

Address 0xBASE+0xE000084 (IPU_SDMA_EVENT_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_NFACK_SDMA_EN_52	IDMAC_NFACK_SDMA_EN_51	IDMAC_NFACK_SDMA_EN_50	IDMAC_NFACK_SDMA_EN_49	IDMAC_NFACK_SDMA_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFACK_SDMA_EN_47	IDMAC_NFACK_SDMA_EN_46	IDMAC_NFACK_SDMA_EN_45	IDMAC_NFACK_SDMA_EN_44	IDMAC_NFACK_SDMA_EN_43	IDMAC_NFACK_SDMA_EN_42	IDMAC_NFACK_SDMA_EN_41	IDMAC_NFACK_SDMA_EN_40	0	0	0	0	0	0	IDMAC_NFACK_SDMA_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-29. SDMA Event Control Register 4 (IPU_SDMA_EVENT_4)

Table 42-26. IPU_SDMA_EVENT_4 Field Descriptions

Field	Description
31–0 *_NFACK_SDMA_EN[1]	Enable New Frame Acknowledge of Channel SDMA event. This bit is the control of New Frame Acknowledge of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.25 SDMA Event Control Register 7 (IPU_SDMA_EVENT_7)

This register contains part of IPUv3EX SDMA events controls. The controls of EOS (End of Scroll) of DMA Channels SDMA events[31:0] can be found in this register.

Address 0xBASE+0xE000088 (IPU_SDMA_EVENT_7) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOS_SDMA_EN_31	0	IDMAC_EOS_SDMA_EN_29	IDMAC_EOS_SDMA_EN_28	IDMAC_EOS_SDMA_EN_27	0	0	IDMAC_EOS_SDMA_EN_24	IDMAC_EOS_SDMA_EN_23	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-30. SDMA Event Control Register 7 (IPU_SDMA_EVENT_7)

Table 42-27. IPU_SDMA_EVENT_7 Field Descriptions

Field	Description
31–0 *_EOS_SDMA_EN[1]	Enable End of Scroll of Channel SDMA event. This bit is the control of End Of Scroll of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.26 SDMA Event Control Register 8 (IPU_SDMA_EVENT_8)

This register contains part of IPUv3EX SDMA events controls. The controls of EOS (End of Scroll) of DMA Channels SDMA events [63:32] can be found in this register.

Address 0xBASE+0xE00008C (IPU_SDMA_EVENT_8) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOS_SDMA_EN_52	IDMAC_EOS_SDMA_EN_51	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOS_SDMA_EN_44	IDMAC_EOS_SDMA_EN_43	IDMAC_EOS_SDMA_EN_42	IDMAC_EOS_SDMA_EN_41	0	0	0	0	0	0	0	IDMAC_EOS_SDMA_EN_32	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-31. SDMA Event Control Register 8 (IPU_SDMA_EVENT_8)

Table 42-28. IPU_SDMA_EVENT_8 Field Descriptions

Field	Description
31–0 *_EOS_SDMA_EN[1]	Enable End of Scroll of Channel SDMA event. This bit is the control of End Of Scroll of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.27 SDMA Event Control Register 11 (IPU_SDMA_EVENT_11)

This register contains part of IPUv3EX SDMA events controls. The controls of EOBND (End of Band) of DMA Channels SDMA events[31:0] can be found in this register.

Address 0xBASE+0xE000090 (IPU_SDMA_EVENT_11) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	IDMAC_EOBND_SDMA_EN_22	IDMAC_EOBND_SDMA_EN_21	IDMAC_EOBND_SDMA_EN_20	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOBND_SDMA_EN_12	IDMAC_EOBND_SDMA_EN_11	0	0	0	0	0	IDMAC_EOBND_SDMA_EN_5	0	IDMAC_EOBND_SDMA_EN_3	IDMAC_EOBND_SDMA_EN_2	IDMAC_EOBND_SDMA_EN_1	IDMAC_EOBND_SDMA_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-32. SDMA Event Control Register 11 (IPU_SDMA_EVENT_11)

Table 42-29. IPU_SDMA_EVENT_11 Field Descriptions

Field	Description
31–0 *_EOBND_SDMA_EN[1]	Enable End of Band of Channel SDMA event. This bit is the control of End Of Band of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.28 SDMA Event Control Register 12 (IPU_SDMA_EVENT_12)

This register contains part of IPUv3EX SDMA events controls. The controls of EOBND (End of Band) of DMA Channels SDMA events [63:32] can be found in this register.

Address 0xBASE+0xE000094 (IPU_SDMA_EVENT_12) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOBND_SDMA_EN_50	IDMAC_EOBND_SDMA_EN_49	IDMAC_EOBND_SDMA_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOBND_SDMA_EN_47	IDMAC_EOBND_SDMA_EN_46	IDMAC_EOBND_SDMA_EN_45	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-33. SDMA Event Control Register 12 (IPU_SDMA_EVENT_12)

Table 42-30. IPU_SDMA_EVENT_12 Field Descriptions

Field	Description
31–0 *_EOBND_SDMA_EN[1]	Enable End of Band of Channel SDMA event. This bit is the control of End Of Band of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.29 SDMA Event Control Register 13 (IPU_SDMA_EVENT_13)

This register contains part of IPUv3EX SDMA events controls. The controls of TH (Threshold) of DMA Channels SDMA events[31:0] can be found in this register.

Address 0xBASE+0xE000098 (IPU_SDMA_EVENT_13) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_TH_SDMA_EN_31	0	IDMAC_TH_SDMA_EN_29	IDMAC_TH_SDMA_EN_28	IDMAC_TH_SDMA_EN_27	0	0	IDMAC_TH_SDMA_EN_24	IDMAC_TH_SDMA_EN_23	IDMAC_TH_SDMA_EN_22	IDMAC_TH_SDMA_EN_21	IDMAC_TH_SDMA_EN_20	0	IDMAC_TH_SDMA_EN_18	IDMAC_TH_SDMA_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_TH_SDMA_EN_15	IDMAC_TH_SDMA_EN_14	IDMAC_TH_SDMA_EN_13	IDMAC_TH_SDMA_EN_12	IDMAC_TH_SDMA_EN_11	IDMAC_TH_SDMA_EN_10	IDMAC_TH_SDMA_EN_9	IDMAC_TH_SDMA_EN_8	IDMAC_TH_SDMA_EN_7	IDMAC_TH_SDMA_EN_6	IDMAC_TH_SDMA_EN_5	IDMAC_TH_SDMA_EN_4	IDMAC_TH_SDMA_EN_3	IDMAC_TH_SDMA_EN_2	IDMAC_TH_SDMA_EN_1	IDMAC_TH_SDMA_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-34. SDMA Event Control Register 13 (IPU_SDMA_EVENT_13)

Table 42-31. IPU_SDMA_EVENT_13 Field Descriptions

Field	Description
31–0 *_TH_SDMA_EN[1]	Enable Threshold of Channel SDMA event. This bit is the control of Threshold of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.30 SDMA Event Control Register 14 (IPU_SDMA_EVENT_14)

This register contains part of IPUv3EX SDMA events controls. The controls of TH (Threshold) of DMA Channels SDMA events [63:32] can be found in this register.

Address 0xBASE+0xE00009C (IPU_SDMA_EVENT_14) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_TH_SDMA_EN_52	IDMAC_TH_SDMA_EN_51	IDMAC_TH_SDMA_EN_50	IDMAC_TH_SDMA_EN_49	IDMAC_TH_SDMA_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_TH_SDMA_EN_47	IDMAC_TH_SDMA_EN_46	IDMAC_TH_SDMA_EN_45	IDMAC_TH_SDMA_EN_44	IDMAC_TH_SDMA_EN_43	IDMAC_TH_SDMA_EN_42	IDMAC_TH_SDMA_EN_41	IDMAC_TH_SDMA_EN_40	0	0	0	0	0	0	IDMAC_TH_SDMA_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-35. SDMA Event Control Register 14 (IPU_SDMA_EVENT_14)

Table 42-32. IPU_SDMA_EVENT_14 Field Descriptions

Field	Description
31–0 *_TH_SDMA_EN[1]	Threshold of Channel SDMA event. This bit is the control of Threshold of Channel #n. 0 SDMA event is disabled. 1 SDMA event is enabled.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.31 IPU Shadow Registers Memory Priority 1 Register (IPU_SRM_PRI1)

The register controls the priority of SRM updates. The priority level for each module that has a shadow of its registers in the SRM should be unique. The priority level defines the order of SRM updates. a module with priority set to 010 will be updated before a module with priority set to 001.

Address 0xBASE+0xE0000A0 (IPU_SRM_PRI1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	ISP_SRM_MODE			ISP_SRM_PRI	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	CSI0_SRM_MODE		CSI0_SRM_PRI			0	0	0	CSI1_SRM_MODE			CSI1_SRM_PRI	
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Figure 42-36. IPU Shadow Registers Memory Priority 1 Register (IPU_SRM_PRI1)

Table 42-33. IPU_SRM_PRI1 Field Descriptions

Field	Description
31-21	Reserved
19-20	
18-16	
15-13	Reserved
12-11 CSI0_SRM_MODE	CSI0 SRM Mode This field controls the SRM logic that handles the CSI0 registers 00 - Automatic swapping is disabled; MCU is allowed to access the CSI1's region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
10-8 CSI0_SRM_PRI	CSI0 SRM priority This bits define the priority of the CSI1 module
7-5	Reserved

Table 42-33. IPU_SRM_PRI1 Field Descriptions (continued)

Field	Description
4-3 CSI1_SRM_MODE	CSI1 SRM Mode This field controls the SRM logic that handles the CSI1 registers 00 - Automatic swapping is disabled; MCU is allowed to access the CSI0's region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
2-0 CSI1_SRM_PRI	CSI1 SRM priority This bits define the priority of the CSI0 module

42.2.3.1.32 IPU Shadow Registers Memory Priority 2 Register (IPU_SRM_PRI2)

The register controls the priority of SRM updates. The priority level for each module that has a shadow of its registers in the SRM should be unique. The priority level defines the order of SRM updates. a module with priority set to 010 will be updated before a module with priority set to 001.

Address 0xBASE+0xE0000A4 (IPU_SRM_PRI2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DI1_SRM_MODE		DI1_SRM_PRI			0	0	0	DI0_SRM_MODE		DI0_SRM_PRI		
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DC_6_SRM_MODE		DC_2_SRM_MODE		DC_SRM_PRI				DP_A1_SRM_MODE		DP_A0_SRM_MODE		DP_S_SRM_MODE		DP_SRM_PRI	
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1

Figure 42-37. IPU Shadow Registers Memory Priority 2 Register (IPU_SRM_PRI2)

Table 42-34. IPU_SRM_PRI2 Field Descriptions

Field	Description
31-29	Reserved
28-27 DI1_SRM_MODE	DCI1 SRM Mode This field controls the SRM logic that handles the DI1 registers 00 - Automatic swapping is disabled; MCU is allowed to access the DI1 region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
26-24 DI1_SRM_PRI	DI1 SRM priority This bits define the priority of the DI1 module
23-21	Reserved
20-19 DI0_SRM_MCU_USE	DI0 SRM is used by MCU This bit indicates that the registers of the DI0 are currently being updated by the MCU. The MCU should set this bit before accessing the SRM part that is relevant to the DI0. The MCU should clear this bit when the update procedure is finished. When this bit is set the SRM mechanism will not update the DI0's registers to avoid data coherency problems. 1 DI0 SRM is currently updated by the MCU 0 DI0 SRM s currently not updated by the MCU
18-16 DI0_SRM_PRI	DI0 SRM priority This bits define the priority of the DI0 module
15-14 DC_6_SRM_MODE	DC Group #6 SRM Mode This field controls the SRM logic that handles the DC Group #6 registers 00 - Automatic swapping is disabled; MCU is allowed to access the DC Group #6's region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
13-12 DC_2_SRM_MODE	DC Group #2 SRM Mode This field controls the SRM logic that handles the DC Group #2 registers 00 - Automatic swapping is disabled; MCU is allowed to access the DC Group #2's region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
11-9 DC_SRM_PRI	DC SRM priority This bits define the priority of the DC module
7-5	Reserved

Table 42-34. IPU_SRM_PRI2 Field Descriptions (continued)

Field	Description
8-7 DP_A1_SRM_MODE	DP Async flow #1 SRM Mode This field controls the SRM logic that handles the DP Async flow #1 registers 00 - Automatic swapping is disabled; MCU is allowed to access the DP Async flow #1 region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
6-5 DP_A0_SRM_MODE	DP Async flow #0 SRM Mode This field controls the SRM logic that handles the DP Async flow #0 registers 00 - Automatic swapping is disabled; MCU is allowed to access the DP Async flow #0 region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame. 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
4-3 DP_S_SRM_MODE	DP sync flow SRM Mode This field controls the SRM logic that handles the DP sync flow registers 00 - Automatic swapping is disabled; MCU is allowed to access the DP sync flow region in the RAM 01 - The SRM logic is controlled by the FSU. The update will be done on the next frame. 10 - Reserved 11 - Update now. The SRM is controlled by the MCU. The Register will be update now
2-0 DP_SRM_PRI	DP SRM priority This bits define the priority of the DP module

42.2.3.1.33 FSU Processing Flow 1 Register (IPU_FS_PROC_FLOW1)

This register contain controls for IPUv3EX’s tasks.

Address 0xBASE+0xE0000A8 (IPU_FS_PROC_FLOW1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VF_IN_VALID	ENC_IN_VALID	VDI_SRC_SEL		PRP_SRC_SEL				ISP_SRC_SEL				PP_ROT_SRC_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PP_SRC_SEL				PRPVF_ROT_SRC_SEL				ALT_ISP_SRC_SEL				PRPENC_ROT_SRC_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-38. FSU Processing Flow 1 Register (IPU_FS_PROC_FLOW1)

Figure 42-39.

Table 42-35. IPU_FS_PROC_FLOW1 Field Descriptions

Field	Description
31 VF_IN_VALID	View-finder input valid. Setting this bit indicates that the buffer in memory for viewfinder is validated by the MCU (valid only when RWS_EN is ‘1’). 0 View-finder should skip buffer in memory. 1 View-finder should use buffer in memory.
30 ENC_IN_VALID	Encoding input valid. Setting this bit indicates that the buffer in memory for encoding is validated by the MCU (valid only when RWS_EN is ‘1’). 0 Encoding should skip buffer in memory. 1 Encoding should use buffer in memory.
29-28 VDI_SRC_SEL	Source select for the VDI 00 MCU 01 CSI direct (cb7) 10 Reserved 11 Reserved

Table 42-35. IPU_FS_PROC_FLOW1 Field Descriptions (continued)

Field	Description
27-24 PRP_SRC_SEL	Source select for the Pre Processing Task 0000 MCU 0001 capture0 (smfc0) 0010 capture1 (smfc1) 0011 capture2 (smfc2) 0100 capture3 (smfc3) 0101 IC direct (cb7) 0110 IRT Encoding 0111 IRT viewfinder 1000 Reserved 1001 Reserved 1010 Reserved 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
23-20	
19-16 PP_ROT_SRC_SEL	Source select for the pre processing task of the IRT (CH 50) 0000 MCU 0001 capture0 (smfc0) 0010 Reserved 0011 capture2 (smfc2) 0100 Reserved 0101 Post-processing 0110 Reserved 0111 Reserved 1000 1001 1010 Reserved 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2

Table 42-35. IPU_FS_PROC_FLOW1 Field Descriptions (continued)

Field	Description
15-12 PP_SRC_SEL	Source select for the pre processing task of the IC 0000 MCU 0001 capture0 (smfc0) 0010 Reserved 0011 capture2 (smfc2) 0100 Reserved 0101 Reserved 0110 Rotation for post-processing 0111 Reserved 1000 1001 1010 Reserved 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
11-8 PRPVF_ROT_SRC_SEL	Source select for the view finder task of the IRT 0000 MCU 0001 capture0 (smfc0) 0010 capture1 (smfc1) 0011 capture2 (smfc2) 0100 capture3 (smfc3) 0101 IC direct (cb7) 0110 Reserved 0111 Reserved 1000 View-finder 1001 Reserved 1010 Reserved 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2

Table 42-35. IPU_FS_PROC_FLOW1 Field Descriptions (continued)

Field	Description
7-4	
3-0 PRPENC_ROT_SRC_SEL	Source select for the encoding task of the IRT 0000 MCU 0001 capture0 (smfc0) 0010 capture1 (smfc1) 0011 capture2 (smfc2) 0100 capture3 (smfc3) 0101 IC direct (cb7) 0110 Reserved 0111 encoding 1000 Reserved 1001 Reserved 1010 Reserved 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2

42.2.3.1.34 FSU Processing Flow 2 Register (IPU_FS_PROC_FLOW2)

This register contains controls for IPUv3EX's tasks.

Address: 0xBASE+0xE0000AC (IPU_FS_PROC_FLOW2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRP_ALT_DEST_SEL				PRP_DEST_SEL				PRPENC_ROT_DEST_SEL				PP_ROT_DEST_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PP_DEST_SEL				PRPVF_ROT_DEST_SEL				PRPVF_DEST_SEL				PRP_ENC_DEST_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-40. FSU Processing Flow 2 Register (IPU_FS_PROC_FLOW2)

Table 42-36. IPU_FS_PROC_FLOW2 Field Descriptions

Field	Description
31-28 PRP_ALT_DEST_SEL	Pre processing alternate flow's destination select (for channel DMAIC_7) 0000 - MCU 0001 - IC input buffer (ch12) 0010 - PP (ch11) 0011 - PP_ROT (ch47) 0100 - DC1 (ch28) 0101 - DC2 (ch41) 0110 - DP_ASYNC1 (ch24) 0111 - DP_ASYNC0 (ch29) 1000 - DP_SYNC1 (ch27) 1001 - DP_SYNC0 (ch23) 1010 - Alt DC2 (ch41) 1011 - Alt DP_ASYNC1 (ch24) 1100 - Alt DP_ASYNC0 (ch29) 1111 - Reserved
27-24 PRP_DEST_SEL	Pre processing destination select (for channel DMAIC_7) 0000 - MCU 0001 - IC input buffer (ch12) 0010 - PP (ch11) 0011 - PP_ROT (ch47) 0100 - DC1 (ch28) 0101 - DC2 (ch41) 0110 - DP_ASYNC1 (ch24) 0111 - DP_ASYNC0 (ch29) 1000 - DP_SYNC1 (ch27) 1001 - DP_SYNC0 (ch23) 1010 - Alt DC2 (ch41) 1011 - Alt DP_ASYNC1 (ch24) 1100 - Alt DP_ASYNC0 (ch29) 1111 - Reserved
23-20 PRPENC_ROT_DEST_SEL	Destination select for Rotation task coming from the Encoding input 0000 MCU 0001 Reserved 0010 Reserved 0011 0100 0101 IC Pre Processing 0110 Reserved 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)

Table 42-36. IPU_FS_PROC_FLOW2 Field Descriptions (continued)

Field	Description
19-16 PP_ROT_DEST_SEL	Destination select for Rotation task coming from the Post Processing input 0000 MCU 0001 Reserved 0010 Reserved 0011 Reserved 0100 IC Playback (Post Processing) 0101 0110 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)
15-12 PP_DEST_SEL	Destination select for post processing task 0000 MCU 0001 Reserved 0010 Reserved 0011 IRT playback 0100 Reserved 0101 Reserved 0110 Reserved 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)
11-8 PRPVF_ROT_DEST_SEL	Destination select for Rotation task coming from the View finder input 0000 MCU 0001 Reserved 0010 Reserved 0011 0100 0101 IC Pre Processing 0110 Reserved 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)

Table 42-36. IPU_FS_PROC_FLOW2 Field Descriptions (continued)

Field	Description
7-4 PRPVF_DEST_SEL	Destination select for View finder task 0000 MCU 0001 IRT viewfinder 0010 Reserved 0011 Reserved 0100 Reserved 0101 Reserved 0110 Reserved 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)
3-0 PRP_ENC_DEST_SEL	Destination select for Encoding task 0000 MCU 0001 IRT Encoding 0010 Reserved 0011 Reserved 0100 Reserved 0101Reserved 0110 Reserved 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)

42.2.3.1.35 IPU FSU Processing Flow 3 Register (IPU_FS_PROC_FLOW3)

This register contains controls for IPUv3EX’s tasks.

Address 0xBASE+0xE0000B0 (IPU_FS_PROC_FLOW3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	EXT_SRC 2_DEST_ SEL		EXT_SRC 1_DEST_ SEL		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SMFC3_DEST_S EL			SMFC2_DEST_SEL				SMFC1_DEST_S EL			SMFC0_DEST_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-41. IPU FSU Processing Flow 3 Register (IPU_FS_PROC_FLOW3)

Table 42-37. IPU_FS_PROC_FLOW3 Field Descriptions

Field	Description
31-24	Reserved
23-22 EXT_SRC2_DEST_SEL	Destination select for External Source 2 00 disabled 01 DP_SYNC0 (ch23) 10 DP_SYNC1 (ch27) 11 DC1 (ch28)
21-20 EXT_SRC1_DEST_SEL	Destination select for External Source 1 00 disabled 01 DP_SYNC0 (ch23) 10 DP_SYNC1 (ch27) 11 DC1 (ch28)
19-14	Reserved
13-11 SMFC3_DEST_SEL	Destination select for SMFC3 000 MCU 001 IRT Encoding 010 IRT viewfinder 011 IRT playback 100 IC Playback (Post Processing) 101 IC Pre Processing 110 111 Reserved

Table 42-37. IPU_FS_PROC_FLOW3 Field Descriptions (continued)

Field	Description
10-7 SMFC2_DEST_SEL	Destination select for SMFC2 0000 MCU 0001 IRT Encoding 0010 IRT viewfinder 0011 IRT playback 0100 IC Playback (Post Processing) 0101 IC Pre Processing 0110 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)
6-4 SMFC1_DEST_SEL	Destination select for SMFC1 000 MCU 001 IRT Encoding 010 IRT viewfinder 011 IRT playback 100 IC Playback (Post Processing) 101 IC Pre Processing 110 111 Reserved
3-0 SMFC0_DEST_SEL	Destination select for SMFC0 0000 MCU 0001 IRT Encoding 0010 IRT viewfinder 0011 IRT playback 0100 IC Playback (Post Processing) 0101 IC Pre Processing 0110 0111 DC1 (ch28) 1000 DC2 (ch41) 1001 DP_SYNC0 (ch23) 1010 DP_SYNC1 (ch27) 1011 DP_ASYNC1 (ch24) 1100 DP_ASYNC0 (ch29) 1101 Alt DC2 (ch41) 1110 Alt DP_ASYNC1 (ch24) 1111 Alt DP_ASYNC0 (ch29)

42.2.3.1.36 IPU FSU Displaying Flow 1 Register (IPU_FS_DISP_FLOW1)

This register contains controls for IPUv3EX’s tasks.

Address 0xBASE+0xE0000B4 (IPU_FS_DISP_FLOW1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	DC1_SRC_SEL				DC2_SRC_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DP_ASYNC1_SRC_SEL				DP_ASYNC0_SRC_SEL				DP_SYNC1_SRC_SEL				DP_SYNC0_SRC_SEL			
W														—		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-42. IPU FSU Processing Flow 3 Register (IPU_FS_PROC_FLOW3)

Table 42-38. IPU_FS_DISP_FLOW1 Field Descriptions

Field	Description
23-20 DC1_SRC_SEL	Source select for DS1/DS2 - MG (graphics) plane (ch28) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001ISP 1010 1011 autoref 1100 autoref+snoop1 1101 External source #1 (e.g. an external module like GPU) 1110 snoop1 1111 External source #2 (e.g. an external module like GPU)
19-16 DC2_SRC_SEL	Source select for DS3 (ch41) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001ISP 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
15-12 DP_ASYNC1_SRC_SEL	Source select for DS1/DS2 - Vx (video) plane (ch24) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2

Table 42-38. IPU_FS_DISP_FLOW1 Field Descriptions (continued)

Field	Description
11-8 DP_ASYNC0_SRC_SEL	Source select for DS2 - MG (graphics) plane (ch29) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
7-4 DP_SYNC1_SRC_SEL	Source select for DS1/DS2 - Vx (video) plane (ch27) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 External source #1 (e.g. an external module like GPU) 1100 External source #2 (e.g. an external module like GPU) 1101 Reserved 1110 snoop1 1111 snoop2
3-0 DP_SYNC0_SRC_SEL	Source select for DS2 - MG (graphics) plane (ch23) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 External source #1 (e.g. an external module like GPU) 1100 External source #2 (e.g. an external module like GPU) 1101 Reserved 1110 snoop1 1111 snoop2

42.2.3.1.37 IPU FSU Displaying Flow 2 Register (IPU_FS_DISP_FLOW2)

This register contains controls for IPUv3EX's tasks.

Address 0xBASE+0xE0000B8 (IPU_FS_DISP_FLOW2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	DC2_ALT_SRC_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DP_ASYNC1_ALT_SRC_SEL				DP_ASYNC0_ALT_SRC_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-43. IPU FSU Displaying Flow 2 Register (IPU_FS_DISP_FLOW2)

Table 42-39. IPU_FS_DISP_FLOW2 Field Descriptions

Field	Description
31-20	Reserved
19-16 DC2_ALT_SRC_SEL	Source select for Alternate DS3 (ch41) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
15-8	Reserved

Table 42-39. IPU_FS_DISP_FLOW2 Field Descriptions (continued)

Field	Description
7-4 DP_ASYNC1_ALT_SRC_SEL	Source select for alternate DS1/DS2 - Vx (video) plane (ch24) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2
3-0 DP_ASYNC0_ALT_SRC_SEL	Source select for alternate DS2 - MG (graphics) plane (ch29) 0000 MCU 0001 capture0 (smfc0) 0010 capture2 (smfc2) 0011 IC encoding 0100 IC viewfinder 0101 IC playback 0110 IRT Encoding 0111 IRT viewfinder 1000 IRT playback 1001 1010 1011 autoref 1100 autoref+snoop1 1101 autoref+snoop2 1110 snoop1 1111 snoop2

42.2.3.1.38 IPU SKIP Register (IPU_SKIP)

This register controls the different frame skipping supported by the IPUv3EX.

Address 0xBASE+0xE0000BC (IPU_SKIP) Access: User read-write

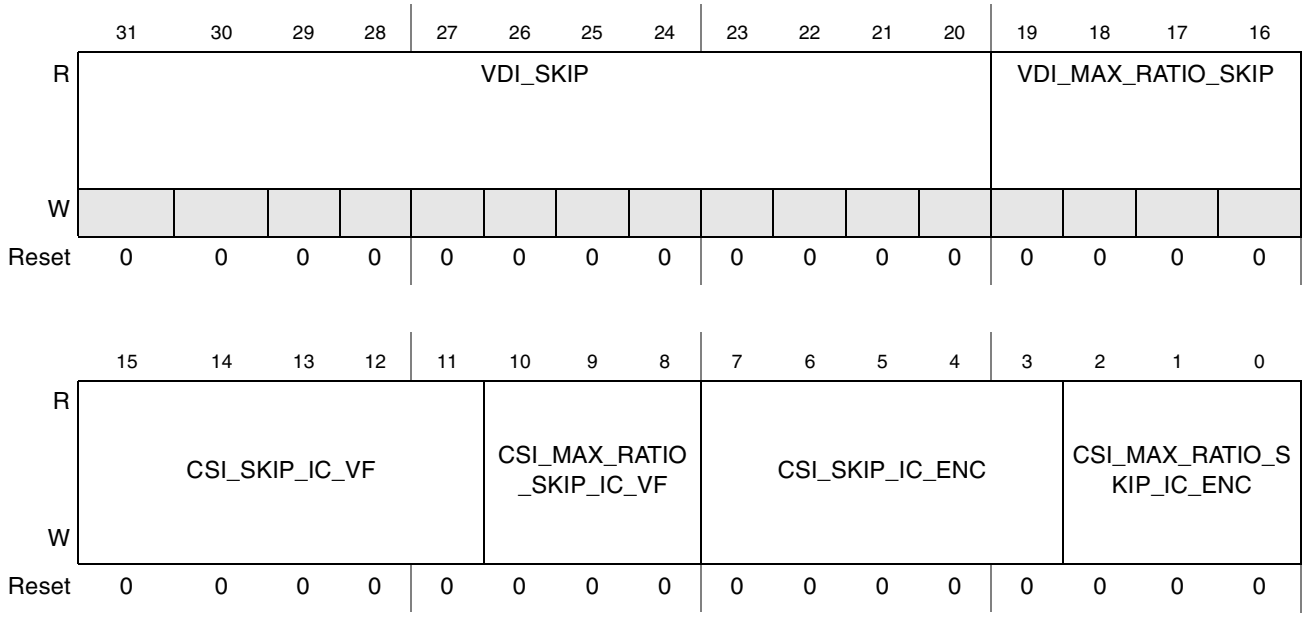


Figure 42-44. IPU SKIP Register (IPU_SKIP)

Table 42-40. IPU_SKIP Field Descriptions

Field	Description
31–20 VDI_SKIP	VDI_SKIP These 12 bits define the skipping pattern of the frames send from the VDI. The VDI avoides reading fields from the memory if the output frame is skipped. Skipping is relevant only if the source to the VDI is coming from the CSI. Skipping is done for a set of frames. The number of frames in a set is defined at VDI_MAX_RATIO_SKIP. when VDI_MAX_RATIO_SKIP = 1 => VDI_SKIP[1:0] is used; other bits are ignored when VDI_MAX_RATIO_SKIP = 2 => VDI_SKIP[2:0] are used; other bits are ignored .. .. when VDI_MAX_RATIO_SKIP = 11 => VDI_SKIP[11:0] are used;
19–16 VDI_MAX_RATIO_SKIP	Maximum Ratio Skip for VDI These bits define the number of frames in a skipping set. The maximum value of this bits is 11. When set to 0 the skipping is disabled.

Table 42-40. IPU_SKIP Field Descriptions (continued)

Field	Description
15–11 CSI_SKIP_IC_VF	CSI SKIP IC_VF These 5 bits define the skipping pattern of the frames send to the IC for view finder task from one of the CSIs as defined on the CSI_SEL and IC_INPUT bits Skipping is done for a set of frames. The number of frames in a set is defined at CSI_MAX_RATIO_SKIP_IC_VF. when CSI_MAX_RATIO_SKIP_IC_VF = 1 => CSI_SKIP_IC_VF[1:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_VF = 2 => CSI_SKIP_IC_VF[2:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_VF =3 => CSI_SKIP_IC_VF[3:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_VF = 4 => CSI_SKIP_IC_VF[4:0] are used; Setting bit #n of CSI_SKIP_IC_VF means that the #n frame in the set is skipped. For example: if CSI_MAX_RATIO_SKIP_IC_VF = 4 and CSI_SKIP_IC_VF = 11010 Frames #0 & Frame #2 will not be skipped as bit0 and bit2 are cleared Frames #1 & Frame #3 will be skipped as bit1 and bit3 are set bit #4 is ignored as CSI_MAX_RATIO_SKIP_IC_VF is set to 4
10–8 CSI_MAX_RATIO_SKIP_IC_VF	CSI Maximum Ratio Skip for IC (view finder task) These bits define the number of frames in a skipping set. The maximum value of this bits is 4. When set to 0 the skipping is disabled.
7–3 CSI_SKIP_IC_ENC	CSI SKIP IC_ENC These 5 bits define the skipping pattern of the frames send to the IC for encoding task from one of the CSIs as defined on the CSI_SEL and IC_INPUT bits Skipping is done for a set of frames. The number of frames in a set is defined at CSI_MAX_RATIO_SKIP_IC_ENC. when CSI_MAX_RATIO_SKIP_IC_ENC = 1 => CSI_SKIP_IC_ENC[1:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_ENC = 2 => CSI_SKIP_IC_ENC[2:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_ENC = 3 => CSI_SKIP_IC_ENC[3:0] are used; other bits are ignored when CSI_MAX_RATIO_SKIP_IC_ENC = 4 => CSI_SKIP_IC_ENC[4:0] are used; Setting bit #n of CSI_SKIP_IC_ENC means that the #n frame in the set is skipped. For example: if CSI_MAX_RATIO_SKIP_IC_ENC = 4 and CSI_SKIP_IC_ENC = 11010 Frames #0 & Frame #2 will not be skipped as bit0 and bit2 are cleared Frames #1 & Frame #3 will be skipped as bit1 and bit3 are set bit #4 is ignored as CSI_MAX_RATIO_SKIP_IC_ENC is set to 4
2–0 CSI_MAX_RATIO_SKIP_IC_ENC	CSI Maximum Ratio Skip for IC (encoding task) These bits define the number of frames in a skipping set. The maximum value of this bits is 4. When set to 0 the skipping is disabled.

42.2.3.1.39 IPU Display Alternate Configuration Register (IPU_DISP_ALT_CONF)

This register controls various parameters that are used for alternate flows related to the display modules.

Address 0xBASE+0xE0000C0 (IPU_DISP_ALT_CONF) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-45. IPU Display Alternate Configuration Register (IPU_DISP_ALT_CONF)

Table 42-41. IPU_DISP_ALT_CONG Field Descriptions

Field	Description
31-0	Reserved

42.2.3.1.40 IPU Display General control Register (IPU_DISP_GEN)

This register controls various aspects of the display port.

Address 0xBASE+0xE0000C4 (IPU_DISP_GEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	DI1_COUNTER_RELEASE	DI0_COUNTER_RELEASE	CSI_VSYNC_DEST	MCU_MAX_BURST_STOP	MCU_T				MCU_DI_ID_9	MCU_DI_ID_8
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	DP_FG_EN_ASYNC1	DP_FG_EN_ASYNC0	DP_ASYNC_DOUBLE_FLOW	DC2_DOUBLE_FLOW	DI1_DUAL_MODE	DI0_DUAL_MODE
W										DP_PIPE_CLR						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-46. IPU Display General control Register (IPU_DISP_GEN)

Table 42-42. DP_DISP_GEN Field Descriptions

Field	Description
31–8	Reserved
25 DI1_COUNTER_RELEASE	DI1 Counter release By default the DI0 counters responsible for waveform generation for sync flow are frozen. For the first attempt to use the DI in sync flow the user should set this bit 1 -counter is released and running 0 - counter is cleared and stopped
24 DI0_COUNTER_RELEASE	DI1 Counter release By default the DI0 counters responsible for waveform generation for sync flow are frozen. For the first attempt to use the DI in sync flow the user should set this bit 1 -counter is released and running 0 - counter is cleared and stopped

Table 42-42. DP_DISP_GEN Field Descriptions (continued)

Field	Description
23 CSI_VSYNC_DEST	CSI_VSYNC destination This bit defines the destination of the VSYNC coming from the CSI's 1 - csi1_vsync is connected to DI0; csi0_vsync is connected to DI1 0 - csi0_vsync is connected to DI0; csi1_vsync is connected to DI1
22 MCU_MAX_BURST_STOP	MCU Maximal burst This bit limit the maximal unspecified length burst. 1 - The maximum unspecified burst length is 8-beat 0 - The unspecified burst length is unlimited
21-18 MCU_T	The address space for accesses through the AHB-lite slave port is 512 MB and it is split internally (with 32MB resolution) according to bits [28:25] of the address. Using the following notation: Address = (IPU_ID[31:29], MSB[28:25], LSB[24:0]) The address is used as follows ("T" is a configurable integer between 0 and 13): MSB<T: access to an external device, with address = (MSB, LSB) T<=MSB<14: access to an external device, with address (MSB-T, LSB)
17 MCU_DI_ID_9	MCU_DI_ID_9 - DI ID via DC channel 9. This bit defines the DI that the MCU DC's access via channel #9 1 MCU accesses DC's channel #9 via DI1. 0 MCU accesses DC's channel #9 via DI0.
16 MCU_DI_ID_8	MCU_DI_ID_8 - DI ID via DC channel 8. This bit defines the DI that the MCU DC's access via channel #8 1 MCU accesses DC's channel #8 via DI1. 0 MCU accesses DC's channel #8 via DI0.
15–7	Reserved
6 DP_PIPE_CLR	DP Pipe Clear This bit clears the internal pipe of the DP. The user may use this bit in case of an error condition This is a self clear bit 1 - Clear the internal pipe of the DP 0 - Idle - does nothing
5 DP_FG_EN_ASYNC1	FG_EN - partial plane Enable for async flow 1. This bit enables the partial plane channel. 1 partial plane channel is enabled. 0 partial plane channel is disabled.
4 DP_FG_EN_ASYNC0	FG_EN - partial plane Enable for async flow 0. This bit enables the partial plane channel. 1 partial plane channel is enabled. 0 partial plane channel is disabled.
3 DP_ASYNC_DOUBLE_FLOW	DP Async Double Flow. This bit define how many async flows are currently handles via DP channel (ch24+29) 1 2 flows are handled via DP 0 single flow is handled via DP

Table 42-42. DP_DISP_GEN Field Descriptions (continued)

Field	Description
2 DC2_DOUBLE_FLOW	DC2 Double Flow. This bit define how many flows are currently handles via DC2 channel (ch41) 11 2 flows are handled via DC2 0 single flow is handled via DC2
1 DI1_DUAL_MODE	DI1 dual mode control 1 DI1 operates in dual mode 0 DI1 is not in dual mode
0 DI0_DUAL_MODE	DI0 dual mode control 1 DI0 operates in dual mode 0 DI0 is not in dual mode

42.2.3.1.41 Display Alternate flow control Register 1 (IPU_DISP_ALT1)

This register controls various aspects of the display port.

Address 0xBASE+0xE0000C8 (IPU_DISP_ALT1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	sel_alt_0				step_repeat_alt_0											
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	cnt_auto_reload_alt_0	cnt_clr_sel_alt_0			run_value_m1_alt_0											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-47. Display Alternate flow control Register 1 (IPU_DISP_ALT1)

Table 42-43. IPU_DISP_ALT1 Field Descriptions

Field	Description
31–28 sel_alt_0	Select alternative parameters instead of DI Sync Wave Gen counter#. The DI is selected according to DP's synchronous channel destination 0000-disable 0001-instead of counter 1 0010-instead of counter 2 ..... 1000-instead of counter 8
27–16 step_repeat_alt_0	This fields defines the amount of repetitions that will be performed by the counter

Table 42-43. IPU_DISP_ALT1 Field Descriptions (continued)

Field	Description
15 cnt_auto_reload_alt_0	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the step_repeat_alt_0 field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the step_repeat_alt_0 field
14–12 cnt_clr_sel_alt_0	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–0 run_value_m1_alt_0	Counter pre defined value This fields defines the counter pre defines value. real value- 1

42.2.3.1.42 Display Alternate flow control Register 2 (IPU_DISP_ALT2)

This register controls various aspects of the display port.

Address 0xBASE+0xE0000CC (IPU_DISP_ALT2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	run_resolution_alt_0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	offset_resolution_alt_0			offset_value_alt_0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-48. Display Alternate flow control Register 2 (IPU_DISP_ALT2)

Table 42-44. IPU_DISP_ALT3 Field Descriptions

Field	Description
31–19	Reserved
18–16 run_resolution_alt_0	Counter Run Resolution This field defines the trigger causing the counter to increment. The counter run resolution should be defined in the same way as in original DI's counter#
15	Reserved
14–12 offset_resolution_alt_0	Counter offset Resolution This field defines the trigger causing the offset counter to increment The counter offset resolution should be defined in the same way as in original DI's counter#
11–0 offset_value_alt_0	Counter offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by

42.2.3.1.43 Display Alternate flow control Register 3 (IPU_DISP_ALT3)

This register controls various aspects of the display port.

Address 0xBASE+0xE0000D0 (IPU_DISP_ALT3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	sel_alt_1				step_repeat_alt_1											
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	cnt_auto_reload_alt_1	cnt_clr_sel_alt_1			run_value_m1_alt_1											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-49. Display Alternate flow control Register 3 (IPU_DISP_ALT3)

Table 42-45. IPU_DISP_ALT3 Field Descriptions

Field	Description
31–28 sel_alt_1	Select alternative parameters instead of DI Sync Wave Gen counter#. The DI is selected according to DP's synchronous channel destination 0000-disable 0001-instead of counter 1 0010-instead of counter 2 ..... 1000-instead of counter 8
27–16 step_repeat_alt_1	This fields defines the amount of repetitions that will be performed by the counter
15 cnt_auto_reload_alt_1	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the step_repeat_alt_0 field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the step_repeat_alt_0 field
14–12 cnt_clr_sel_alt_1	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–0 run_value_m1_alt_1	Counter pre defined value This fields defines the counter pre defines value. real value- 1

42.2.3.1.44 IPU Display Alternate flow control Register 4 (IPU_DISP_ALT4)

This register controls various aspects of the display port.

Address 0xBASE+0xE0000D4 (IPU_DISP_ALT4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	run_resolution_alt_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	offset_resolution_alt_1			offset_value_alt_1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-50. IPU Display Alternate flow control Register 4 (IPU_DISP_ALT4)

Table 42-46. IPU_DISP_ALT4 Field Descriptions

Field	Description
31–19	Reserved
18–16	Counter Run Resolution This field defines the trigger causing the counter to increment. The counter run resolution should be defined in the same way as in original DI's counter#
15	Reserved
14–12	Counter offset Resolution This field defines the trigger causing the offset counter to increment The counter offset resolution should be defined in the same way as in original DI's counter#
11–0	Counter offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by

42.2.3.1.45 IPU Autorefresh and Snooping Control Register (IPU_SNOOP)

This register controls the snooping mechanism

Address 0xBASE+0xE0000D8 (IPU_SNOOP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SNOOP2_SYNC_BYP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	AUTOREF_PER									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-51. IPU Autorefresh and Snooping Control Register (IPU_SNOOP)

Table 42-47. IPU_SNOOP Field Descriptions

Field	Description
31-17	Reserved
16 SNOOP2_SYNC_BYP	This bits control the bypass of the synchronizer on emi_snooping2 signal. This bit is for test purposes only. The user should not set this bit 1- bypass the emi_snooping2 synchronizer 0 - normal mode emi_snooping2 is internally synchronized
15-10	Reserved
9–0 AUTOREF_PER	Autorefresh period minus 1. The actual value of autorefresh period is equal to the high speed clock (HSP_CLK) period multiplied by 2^17*(AUTO_REF_PER+1).

42.2.3.1.46 IPU Memory Reset Control Register (IPU_MEM_RST)

This register controls the memory reset mechanism. IPUv3EX has a hardware mechanism for clearing the content of the internal memories. This allows the user to clear the content of or more of the internal memories without the need to perform write accesses to the memories.

Address 0xBASE+0xE0000DC (IPU_MEM_RST) Access: User read/write

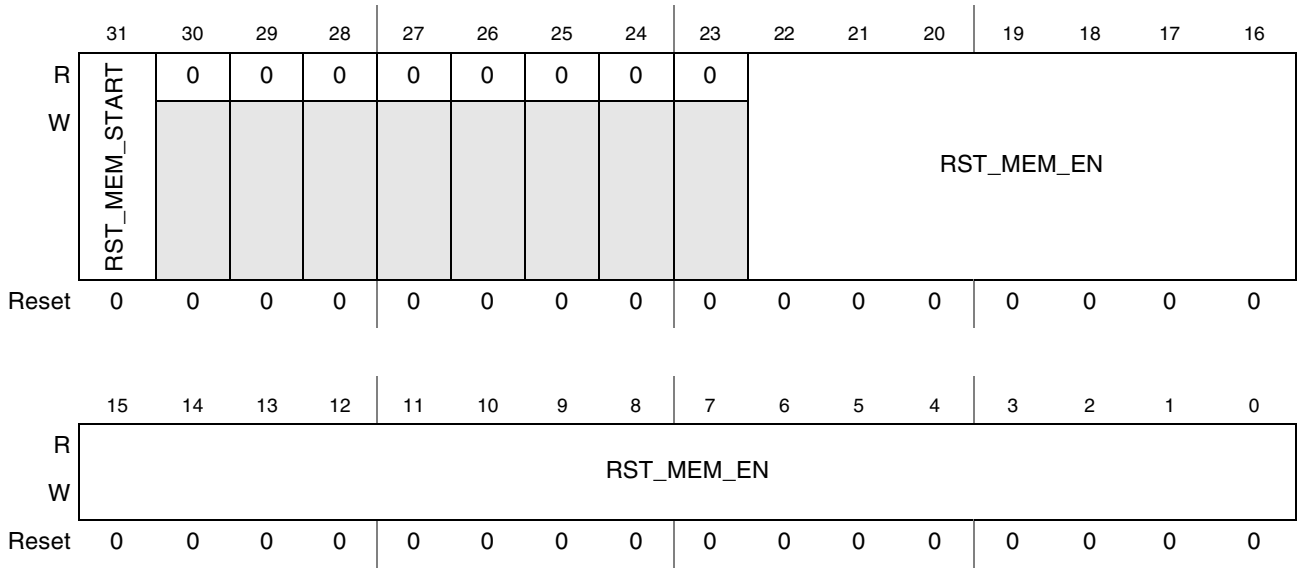


Figure 42-52. IPU Memory Reset Control Register (IPU_MEM_RST)

Table 42-48. IPU_MEM_RST Field Descriptions

Field	Description
31 RST_MEM_START	Memory Reset Start Writing one to this bit activate the memory reset mechanism. The memories that their corresponding RST_MEM_EN bit is set will be cleared. When the memory reset mechanism completes the memory clearing procedure this bit will be automatically cleared. 1 The memory reset mechanism is activated and busy 0 Idle, the memory reset mechanism is not working.
22–0 RST_MEM_EN	Reset Memory Enable Each bit on this field enables the memory reset mechanism for a specific memory. The user should set the relevant bits for the memories that need to be cleared. Below is the list of memories and their corresponding bit. srm = rst_mem_en[0] alpha = rst_mem_en[1] cpmem = rst_mem_en[2] tpm = rst_mem_en[3] mpm = rst_mem_en[4] bm = rst_mem_en[5] rm = rst_mem_en[6] dstm = rst_mem_en[7] dsom = rst_mem_en[8] lut0 = rst_mem_en[9] lut1 = rst_mem_en[10] ram_smfc = rst_mem_en[11] isp_st = rst_mem_en[12] isp_hist= rst_mem_en[13] ram_pp1= rst_mem_en[14] ram_mix1 = rst_mem_en[15] ram_pp0 = rst_mem_en[16] ram_mix0 = rst_mem_en[17] fifo_ram = rst_mem_en[18] isp_tbpr = rst_mem_en[19] dc_template = rst_mem_en[20] dmfc_rd = rst_mem_en[21] dmfc_wr = rst_mem_en[22]

42.2.3.1.47 IPU Power modes Control Register (IPU_PM)

This register controls the automatic transitions of the IPUv3EX between different power modes of the SoC and handles the clock change modes.

Address 0xBASE+0xE0000E0 (IPU_PM) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	LPSR_MODE	DI1_SRM_CLOCK_CHANGE_MODE	DI1_CLK_PERIOD_1						DI1_CLK_PERIOD_0							
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLOCK_MODE_STAT	DI0_SRM_CLOCK_CHANGE_MODE	DI0_CLK_PERIOD_1						DI0_CLK_PERIOD_0							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Figure 42-53. IPU Power modes Control Register (IPU_PM)

Table 42-49. IPU_PM Field Descriptions

Field	Description
31 LPSR_MODE	LPSR Mode This bit indicates that the next attempt for entering low power mode is an attempt to move to LPST mode. Setting this bit by the user is essential in order to assure proper response of the IPUv3EX to the assertion of the stop request from the CCM. 1 - Next low power mode will be LPSR 0 - Next low power mode is not LPSR
30 DI1_SRM_CLOCK_CHANGE_MODE	SRM clock change mode When the clock is going to be changed to any new ratio other then 1:1, 1:2. 1:4. The user needs to prepare an alternate set of DI setting in the SRM. This bit enable this mode. This bit is self cleared. 1 - SRM clock change mode is enabled; the next clock change will be done by updating the DI settings from the SRM 0 - SRM clock change mode is disabled.
29-23 DI1_CLK_PERIOD_1	DI1_CLK period option 1. This parameter defines the period of the clock that the DI1 works with. This parameter contains integer part (bits [6:4]) and fractional part (bits [3:0]). Setting this value to 1.0 (default) means that the DI1 works on the fastest possible clock. Setting a value smaller than 1.0 is not allowed. The value to be programmed to the DI1_CLK_PERIOD_1 field is equal to: Fast_freq/Target_freq Where: Target_freq = The frequency that the DI clock works with Fast_freq = fastest possible clock that the DI can work with
22-16 DI1_CLK_PERIOD_0	DI1_CLK period option 0. This parameter defines the period of the clock that the DI1 works with. This parameter contains integer part (bits [6:4]) and fractional part (bits [3:0]). Setting this value to 1.0 (default) means that the DI1 works on the fastest possible clock. Setting a value smaller than 1.0 is not allowed. The value to be programmed to the DI1_CLK_PERIOD_1 field is equal to: Fast_freq/Target_freq Where: Target_freq = The frequency that the DI clock works with Fast_freq = fastest possible clock that the DI can work with
15 CLCOK_MODE_STAT	Clock mode status This is a read only bit indicating what is the current clock mode 1 - current clock mode is 1 0 - current clock mode is 0
14 DI0_SRM_CLOCK_CHANGE_MODE	SRM clock change mode When the clock is going to be changed to any new ratio other then 1:1, 1:2. 1:4. The user needs to prepare an alternate set of DI setting in the SRM. This bit enable this mode. This bit is self cleared. 1 - SRM clock change mode is enabled; the next clock change will be done by updating the DI settings from the SRM 0 - SRM clock change mode is disabled.

Table 42-49. IPU_PM Field Descriptions (continued)

Field	Description
13-7 DI0_CLK_PERIOD_1	DI0_CLK period option 1. This parameter defines the period of the clock that the DI0 works with. This parameter contains integer part (bits [6:4]) and fractional part (bits [3:0]). Setting this value to 1.0 (default) means that the DI0 works on the fastest possible clock. Setting a value smaller than 1.0 is not allowed. The value to be programmed to the DI0_CLK_PERIOD_1 field is equal to: Fast_freq/Target_freq Where: Target_freq = The frequency that the DI clock works with Fast_freq = fastest possible clock that the DI can work with
6-0 DI0_CLK_PERIOD_0	DI0_CLK period option 0. This parameter defines the period of the clock that the DI0 works with. This parameter contains integer part (bits [6:4]) and fractional part (bits [3:0]). Setting this value to 1.0 (default) means that the DI0 works on the fastest possible clock. Setting a value smaller than 1.0 is not allowed. The value to be programmed to the DI0_CLK_PERIOD_1 field is equal to: Fast_freq/Target_freq Where: Target_freq = The frequency that the DI clock works with Fast_freq = fastest possible clock that the DI can work with

42.2.3.1.48 IPU General Purpose Register (IPU_GPR)

The register contains general purpose bits.

Address 0xBASE+0xE0000E4 (IPU_GPR) Access: User read-write

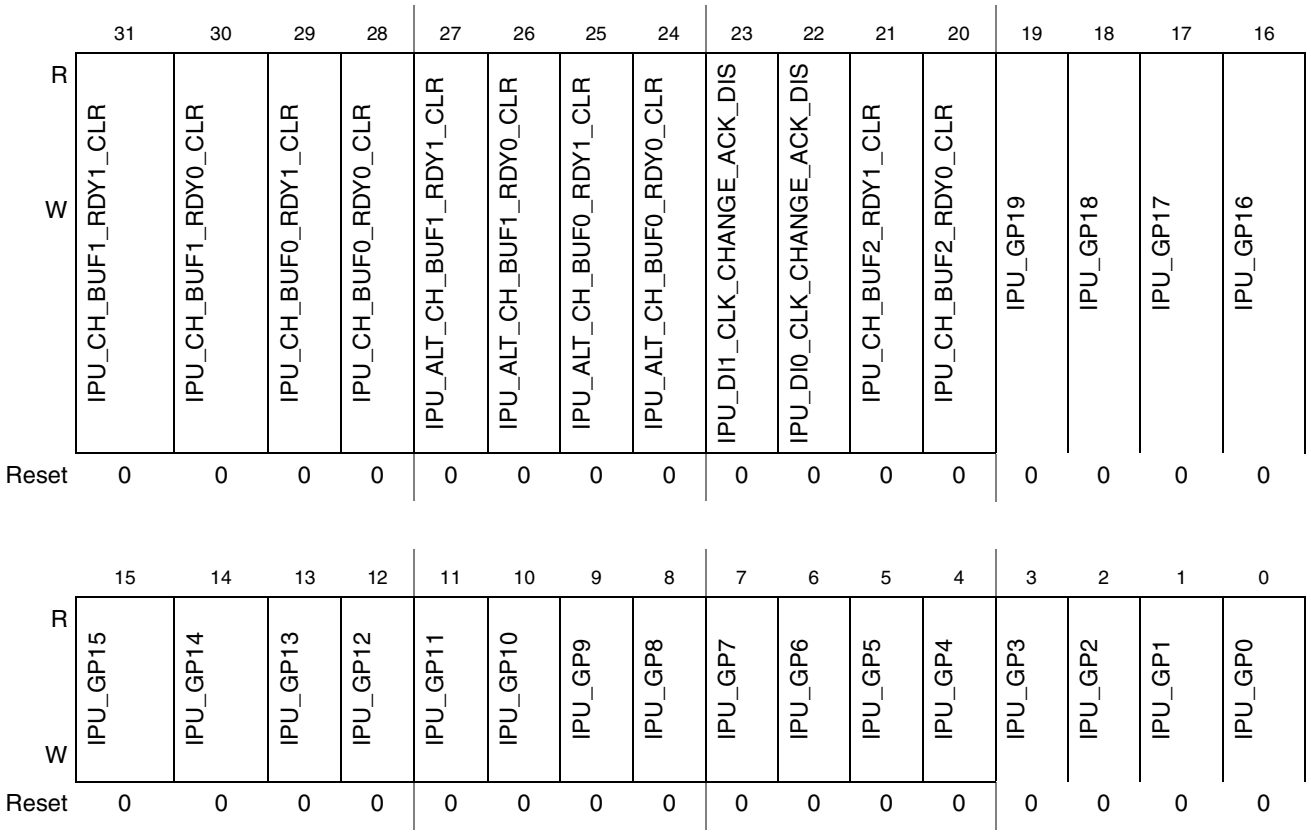


Figure 42-54. IPU General Purpose Register (IPU_GPR)

Table 42-50. IPU_GPR Field Descriptions

Field	Description
31 IPU_CH_BUF1_RDY1_CLR	This bit defines the IPU_CH_BUF1_RDY1 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF1_RDY1 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF1_RDY1 is w1s register
30 IPU_CH_BUF1_RDY0_CLR	This bit defines the IPU_CH_BUF1_RDY0 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF1_RDY0 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF1_RDY0 is w1s register

Table 42-50. IPU_GPR Field Descriptions (continued)

Field	Description
29 IPU_CH_BUF0_RDY1_CLR	This bit defines the IPU_CH_BUF0_RDY1 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF0_RDY1 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF0_RDY1 is w1s register
28 IPU_CH_BUF0_RDY0_CLR	This bit defines the IPU_CH_BUF0_RDY0 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF0_RDY0 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF0_RDY0 is w1s register
27 IPU_ALT_CH_BUF1_RDY1_CLR	This bit defines the IPU_ALT_CH_BUF1_RDY1 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_ALT_CH_BUF1_RDY1 is w1c register 0 writing one to a bit of this register sets this bit IPU_ALT_CH_BUF1_RDY1 is w1s register
26 IPU_ALT_CH_BUF1_RDY0_CLR	This bit defines the IPU_ALT_CH_BUF1_RDY0 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_ALT_CH_BUF1_RDY0 is w1c register 0 writing one to a bit of this register sets this bit IPU_ALT_CH_BUF1_RDY0 is w1s register
25 IPU_ALT_CH_BUF0_RDY1_CLR	This bit defines the IPU_ALT_CH_BUF0_RDY1 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_ALT_CH_BUF0_RDY1 is w1c register 0 writing one to a bit of this register sets this bit IPU_ALT_CH_BUF0_RDY1 is w1s register
24 IPU_ALT_CH_BUF0_RDY0_CLR	This bit defines the IPU_ALT_CH_BUF0_RDY0 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_ALT_CH_BUF0_RDY0 is w1c register 0 writing one to a bit of this register sets this bit IPU_ALT_CH_BUF0_RDY0 is w1s register
23 IPU_DI1_CLK_CHANGE_ACK_DIS	Disable DI1's clock change mechanism. 1- clock change mechanism is disabled. DI automatically acknowledges a clock change request 0 - clock change mechanism is disabled. DI performs the clock change procedure
22 IPU_DI0_CLK_CHANGE_ACK_DIS	Disable DI0's clock change mechanism. 1- clock change mechanism is disabled. DI automatically acknowledges a clock change request 0 - clock change mechanism is disabled. DI performs the clock change procedure

Table 42-50. IPU_GPR Field Descriptions (continued)

Field	Description
21 IPU_CH_BUF2_RDY1_CLR	This bit defines the IPU_CH_BUF2_RDY1 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF2_RDY1 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF2_RDY1 is w1s register
20 IPU_CH_BUF2_RDY0_CLR	This bit defines the IPU_CH_BUF2_RDY0 properties. This register can be a write one to clear OR write one to set. 1 writing one to a bit of this register clears this bit; IPU_CH_BUF2_RDY0 is w1c register 0 writing one to a bit of this register sets this bit IPU_CH_BUF2_RDY0 is w1s register
19–0 IPU_GP<i>[1]	IPUv3EX General Purpose bit. This bits are general Read/Write bits, reserved for future use

[1] Indicates the corresponding DMA channel number.

42.2.3.1.49 IPU Channel Double Buffer Mode Select 0 Register (IPU_CH_DB_MODE_SEL0)

The register contains double buffer mode select control information for 32 IPUv3EX's DMA channels.

Address 0xBASE+0xE000150 (IPU_CH_DB_MODE_SEL_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_CH_DB_MODE_SEL_31	0	DMA_CH_DB_MODE_SEL_29	DMA_CH_DB_MODE_SEL_28	DMA_CH_DB_MODE_SEL_27	0	0	DMA_CH_DB_MODE_SEL_24	DMA_CH_DB_MODE_SEL_23	DMA_CH_DB_MODE_SEL_22	DMA_CH_DB_MODE_SEL_21	DMA_CH_DB_MODE_SEL_20	0	DMA_CH_DB_MODE_SEL_18	DMA_CH_DB_MODE_SEL_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_DB_MODE_SEL_15	DMA_CH_DB_MODE_SEL_14	DMA_CH_DB_MODE_SEL_13	DMA_CH_DB_MODE_SEL_12	DMA_CH_DB_MODE_SEL_11	DMA_CH_DB_MODE_SEL_10	DMA_CH_DB_MODE_SEL_9	DMA_CH_DB_MODE_SEL_8	DMA_CH_DB_MODE_SEL_7	DMA_CH_DB_MODE_SEL_6	DMA_CH_DB_MODE_SEL_5	DMA_CH_DB_MODE_SEL_4	DMA_CH_DB_MODE_SEL_3	DMA_CH_DB_MODE_SEL_2	DMA_CH_DB_MODE_SEL_1	DMA_CH_DB_MODE_SEL_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-55. IPU Channel Double Buffer Mode Select 0 Register (IPU_CH_DB_MODE_SEL0)

Table 42-52. IPU_CH_DB_MODE_SEL0 Field Descriptions

Field	Description
31–0 DMA_CH_DB_MODE_SEL_*[1]	Double buffer mode select. This bit indicates if a double buffer is used for this channel. 0 Double buffer is not used for this channel. 1 Double buffer is used for this channel.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.50 IPU Channel Double Buffer Mode Select 1 Register (IPU_CH_DB_MODE_SEL1)

The register contains double buffer mode select control information for 32 IPUv3EX's DMA channels.

Address 0xBASE+0xE000154 (IPU_CH_DB_MODE_SEL_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_DB_MODE_SEL_52	DMA_CH_DB_MODE_SEL_51	DMA_CH_DB_MODE_SEL_50	DMA_CH_DB_MODE_SEL_49	DMA_CH_DB_MODE_SEL_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_DB_MODE_SEL_47	DMA_CH_DB_MODE_SEL_46	DMA_CH_DB_MODE_SEL_45	DMA_CH_DB_MODE_SEL_44	DMA_CH_DB_MODE_SEL_43	DMA_CH_DB_MODE_SEL_42	DMA_CH_DB_MODE_SEL_41	DMA_CH_DB_MODE_SEL_40	0	0	0	0	0	0	DMA_CH_DB_MODE_SEL_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-56. IPU Channel Double Buffer Mode Select 1 Register (IPU_CH_DB_MODE_SEL1)

Table 42-53. IPU_CH_DB_MODE_SEL_1 Field Descriptions

Field	Description
31–0 DMA_CH_DB_MODE_SEL_*[1]	Double buffer mode select. This bit indicates if a double buffer is used for this channel. 0 Double buffer is not used for this channel. 1 Double buffer is used for this channel.

[1] Indicates the corresponding DMA channel number.

42.2.3.1.51 IPU Alternate Channel Double Buffer Mode Select 0 Register (IPU_ALT_CH_DB_MODE_SEL0)

The register contains double buffer mode select control information for 32 IPUv3EX's DMA channels.

Address 0xBASE+0xE000168 (IPU_ALT_CH_DB_MODE_SEL_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	DMA_CH_ALT_DB_MODE_SEL_29	0	0	0	0	DMA_CH_ALT_DB_MODE_SEL_24	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DMA_CH_ALT_DB_MODE_SEL_7	DMA_CH_ALT_DB_MODE_SEL_6	DMA_CH_ALT_DB_MODE_SEL_5	DMA_CH_ALT_DB_MODE_SEL_4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-57. IPU Alternate Channel Double Buffer Mode Select 0 Register (IPU_ALT_CH_DB_MODE_SEL0)

Table 42-54. IPU_ALT_CH_DB_MODE_SEL0 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_DB_MODE_SEL_*[1]	Double buffer mode select. This bit indicates if a double buffer is used for this channel. 0 Double buffer is not used for this channel. 1 Double buffer is used for this channel.

1 Indicates the corresponding DMA channel number.

42.2.3.1.52 IPU Alternate Channel Double Buffer Mode Select 1 Register (IPU_ALT_CH_DB_MODE_SEL1)

The register contains double buffer mode select control information for 32 IPUv3EX's DMA channels.

Address 0xBASE+0xE00016C (IPU_ALT_CH_DB_MODE_SEL_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_ALT_DB_MODE_SEL_52	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DMA_CH_ALT_DB_MODE_SEL_41	0	0	0	0	0	0	0	DMA_CH_ALT_DB_MODE_SEL_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-58. IPU Alternate Channel Double Buffer Mode Select 1 Register (IPU_ALT_CH_DB_MODE_SEL1)

Table 42-55. IPU_ALT_CH_DB_MODE_SEL_1 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_DB_MODE_SEL_*[1]	Double buffer mode select. This bit indicates if a double buffer is used for this channel. 0 Double buffer is not used for this channel. 1 Double buffer is used for this channel.

1 Indicates the corresponding DMA channel number.

42.2.3.1.53 IPU Channel Triple Buffer Mode Select 0 Register (IPU_CH_TRB_MODE_SEL0)

The register contains triple buffer mode select control information for 32 IPUv3EX's DMA channels.

When the channel is configured for triple buffer mode. The double buffer mode settings configured on the corresponding DB_MODE_SEL bit are overridden.

Address 0xBASE+0xE000178 (IPU_CH_TRB_MODE_SEL_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	DMA_CH_TRB_MODE_SEL_13	0	0	DMA_CH_TRB_MODE_SEL_10	DMA_CH_TRB_MODE_SEL_9	DMA_CH_TRB_MODE_SEL_8	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

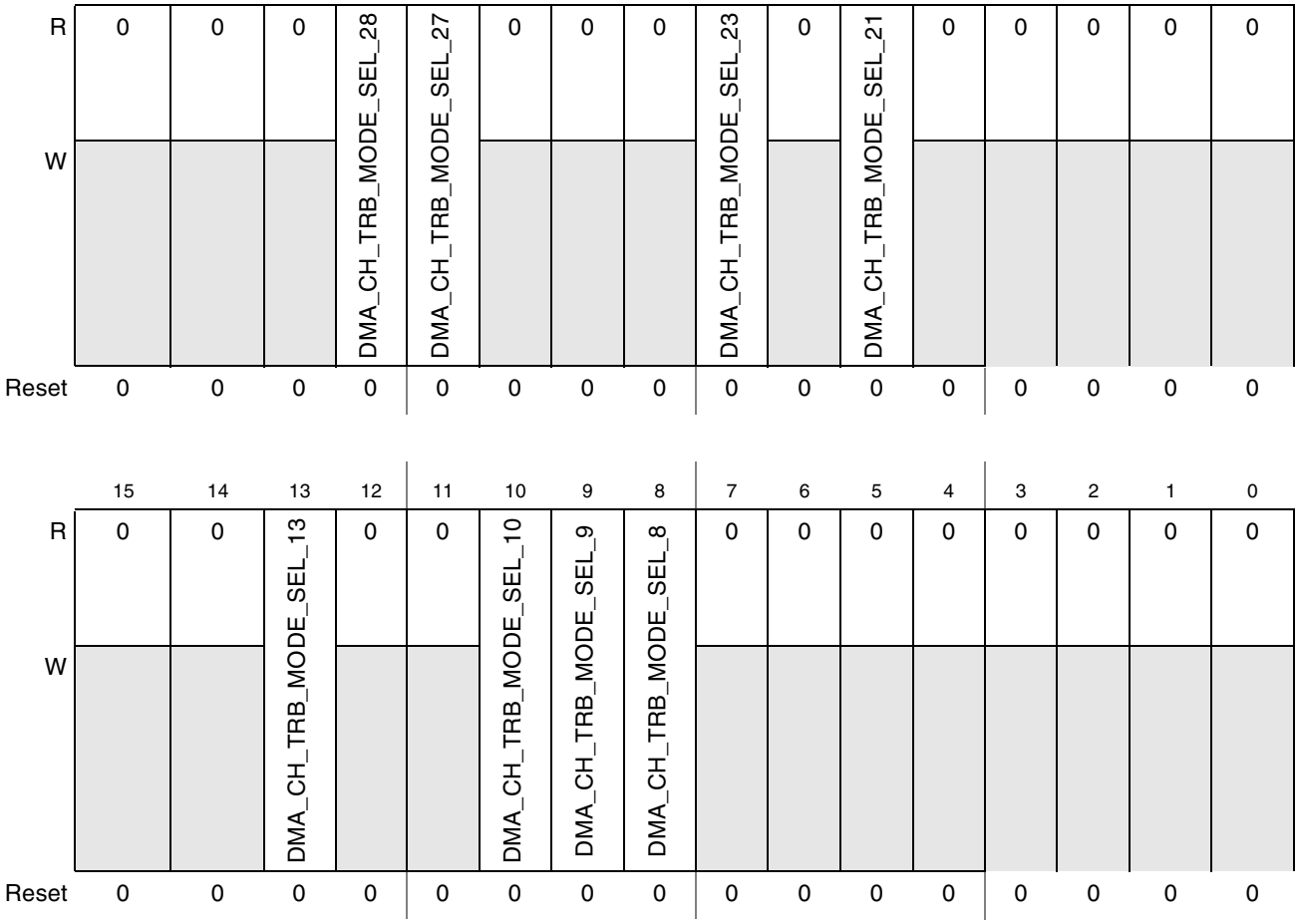


Figure 42-59. IPU Channel Triple Buffer Mode Select 0 Register (IPU_CH_TRB_MODE_SEL0)

Table 42-56. IPU_CH_TRB_MODE_SEL0 Field Descriptions

Field	Description
31–0 DMA_CH_TRB_MODE_SEL_*[1]	Triple buffer mode select. This bit indicates if a triple buffer is used for this channel. 0 Triple buffer is not used for this channel. 1 Triple buffer is used for this channel.

1 Indicates the corresponding DMA channel number.

42.2.3.1.54 IPU Channel Triple Buffer Mode Select 1 Register (IPU_CH_TRB_MODE_SEL1)

The register contains triple buffer mode select control information for 32 IPUv3EX's DMA channels.

When the channel is configured for triple buffer mode. The double buffer mode settings configured on the corresponding DB_MODE_SEL bit are overridden.

Address 0xBASE+0xE00017C (IPU_CH_TRB_MODE_SEL_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-60. IPU Channel Triple Buffer Mode Select 1 Register (IPU_CH_TRB_MODE_SEL1)

Table 42-57. IPU_CH_TRB_MODE_SEL_1 Field Descriptions

Field	Description
31–0 DMA_CH_TRB_MODE_SEL_*[1]	Triple buffer mode select. This bit indicates if a triple buffer is used for this channel. 0 Triple buffer is not used for this channel. 1 Triple buffer is used for this channel.

1 Indicates the corresponding DMA channel number.

42.2.3.2 IPU Status registers

The registers below are IPUv3EX's status registers. These registers are not stored in the SRM during power gating mode.

42.2.3.2.1 IPU Interrupt Status Register 1 (IPU_INT_STAT_1)

This register contains part of IPUv3EX interrupts status bits. The status bits of EOF (end of frame) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000200 (IPU_INT_STAT_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOF_31	0	IDMAC_EOF_29	IDMAC_EOF_28	IDMAC_EOF_27	0	0	IDMAC_EOF_24	IDMAC_EOF_23	IDMAC_EOF_22	IDMAC_EOF_21	IDMAC_EOF_20	0	IDMAC_EOF_18	IDMAC_EOF_17	0
W	w1c		w1c	w1c	w1c			w1c	w1c	w1c	w1c	w1c		w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOF_15	IDMAC_EOF_14	IDMAC_EOF_13	IDMAC_EOF_12	IDMAC_EOF_11	IDMAC_EOF_10	IDMAC_EOF_9	IDMAC_EOF_8	IDMAC_EOF_7	IDMAC_EOF_6	IDMAC_EOF_5	IDMAC_EOF_4	IDMAC_EOF_3	IDMAC_EOF_2	IDMAC_EOF_1	IDMAC_EOF_0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-61. IPU Interrupt Status Register 1 (IPU_INT_STAT_1)

Table 42-58. IPU_INT_STAT_1 Field Descriptions

Field	Description
31–0 *_EOF[1]	Enable End of Frame of Channel interrupt. This bit is the control of End Of Frame of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.2 IPU Interrupt Status Register 2 (IPU_INT_STAT_2)

This register contains part of IPUv3EX interrupts status bits. The status bits of EOF (end of frame) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000204 (IPU_INT_STAT_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOF_52	IDMAC_EOF_51	IDMAC_EOF_50	IDMAC_EOF_49	IDMAC_EOF_48
W												w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOF_47	IDMAC_EOF_46	IDMAC_EOF_45	IDMAC_EOF_44	IDMAC_EOF_43	IDMAC_EOF_42	IDMAC_EOF_41	IDMAC_EOF_40	0	0	0	0	0	0	IDMAC_EOF_33	0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c							w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-62. IPU Interrupt Status Register 2 (IPU_INT_STAT_2)

Table 42-59. IPU_INT_STAT_2 Field Descriptions

Field	Description
31–0 *_EOF[1]	End of Frame of Channel interrupt. This bit is the status bit of End Of Frame of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

1 Indicates the corresponding DMA channel number.

42.2.3.2.3 IPU Interrupt Status Register 3 (IPU_INT_STAT_3)

This register contains part of IPUv3EX interrupts status bits. The status bits of NFACK (New Frame Ack) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000208 (IPU_INT_STAT_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_NFACK_31	0	IDMAC_NFACK_29	IDMAC_NFACK_28	IDMAC_NFACK_27	0	0	IDMAC_NFACK_24	IDMAC_NFACK_23	IDMAC_NFACK_22	IDMAC_NFACK_21	IDMAC_NFACK_20	0	IDMAC_NFACK_18	IDMAC_NFACK_17	0
W	w1c		w1c	w1c	w1c			w1c	w1c	w1c	w1c	w1c		w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFACK_15	IDMAC_NFACK_14	IDMAC_NFACK_13	IDMAC_NFACK_12	IDMAC_NFACK_11	IDMAC_NFACK_10	IDMAC_NFACK_9	IDMAC_NFACK_8	IDMAC_NFACK_7	IDMAC_NFACK_6	IDMAC_NFACK_5	IDMAC_NFACK_4	IDMAC_NFACK_3	IDMAC_NFACK_2	IDMAC_NFACK_1	IDMAC_NFACK_0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-63. IPU Interrupt Status Register 3 (IPU_INT_STAT_3)

Table 42-60. IPU_INT_STAT_3 Field Descriptions

Field	Description
31–0 *_NFACK[1]	Enable New Frame Ack of Channel interrupt. This bit is the status bits of New Frame Ack of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

1 Indicates the corresponding DMA channel number.

42.2.3.2.4 IPU Interrupt Status Register 4 (IPU_INT_STAT_4)

This register contains part of IPUv3EX interrupts status bits. The status bits of NFACK (New Frame Ack) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE00020C (IPU_INT_STAT_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_NFACK_52	IDMAC_NFACK_51	IDMAC_NFACK_50	IDMAC_NFACK_49	IDMAC_NFACK_48
W												w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFACK_47	IDMAC_NFACK_46	IDMAC_NFACK_45	IDMAC_NFACK_44	IDMAC_NFACK_43	IDMAC_NFACK_42	IDMAC_NFACK_41	IDMAC_NFACK_40	0	0	0	0	0	0	IDMAC_NFACK_33	0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c							w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-64. IPU Interrupt Status Register 4 (IPU_INT_STAT_4)

Table 42-61. IPU_INT_STAT_4 Field Descriptions

Field	Description
31–0 *_NFACK[1]	Enable New Frame Ack of Channel interrupt. This bit is the status bit of New Frame Ack of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

1 Indicates the corresponding DMA channel number.

42.2.3.2.5 IPU Interrupt Status Register 5 (IPU_INT_STAT_5)

This register contains part of IPUv3EX interrupts status bits. The status bits of the New-frame before end-of-frame indication (NFB4EOF_ERR) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000210 (IPU_INT_STAT_5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_NFB4EOF_ERR_31	0	IDMAC_NFB4EOF_ERR_29	IDMAC_NFB4EOF_ERR_28	IDMAC_NFB4EOF_ERR_27	0	0	IDMAC_NFB4EOF_ERR_24	IDMAC_NFB4EOF_ERR_23	IDMAC_NFB4EOF_ERR_22	IDMAC_NFB4EOF_ERR_21	IDMAC_NFB4EOF_ERR_20	0	IDMAC_NFB4EOF_ERR_18	IDMAC_NFB4EOF_ERR_17	0
W	w1c		w1c	w1c	w1c			w1c	w1c	w1c	w1c	w1c		w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFB4EOF_ERR_15	IDMAC_NFB4EOF_ERR_14	IDMAC_NFB4EOF_ERR_13	IDMAC_NFB4EOF_ERR_12	IDMAC_NFB4EOF_ERR_11	IDMAC_NFB4EOF_ERR_10	IDMAC_NFB4EOF_ERR_9	IDMAC_NFB4EOF_ERR_8	IDMAC_NFB4EOF_ERR_7	IDMAC_NFB4EOF_ERR_6	IDMAC_NFB4EOF_ERR_5	IDMAC_NFB4EOF_ERR_4	IDMAC_NFB4EOF_ERR_3	IDMAC_NFB4EOF_ERR_2	IDMAC_NFB4EOF_ERR_1	IDMAC_NFB4EOF_ERR_0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-65. IPU Interrupt Status Register 5 (IPU_INT_STAT_5)

Table 42-62. IPU_INT_STAT_5 Field Descriptions

Field	Description
31–0 *_NFB4EOF_ERR[1]	New Frame before end-of-frame error indication of Channel interrupt. This bit is the status bit of New Frame before end-of-frame error interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

1 Indicates the corresponding DMA channel number.

42.2.3.2.6 IPU Interrupt Status Register 6 (IPU_INT_STAT_6)

This register contains part of IPUv3EX interrupts status bits. The status bits of the New-frame before end-of-frame indication (NFB4EOF_ERR) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000214 (IPU_INT_STAT_6) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_NFB4EOF_ERR_52	IDMAC_NFB4EOF_ERR_51	IDMAC_NFB4EOF_ERR_50	IDMAC_NFB4EOF_ERR_49	IDMAC_NFB4EOF_ERR_48
W												w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_NFB4EOF_ERR_47	IDMAC_NFB4EOF_ERR_46	IDMAC_NFB4EOF_ERR_45	IDMAC_NFB4EOF_ERR_44	IDMAC_NFB4EOF_ERR_43	IDMAC_NFB4EOF_ERR_42	IDMAC_NFB4EOF_ERR_41	IDMAC_NFB4EOF_ERR_40	0	0	0	0	0	0	IDMAC_NFB4EOF_ERR_33	0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c							w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-66. IPU Interrupt Status Register 6 (IPU_INT_STAT_6)

Table 42-63. IPU_INT_STAT_6 Field Descriptions

Field	Description
31–0 *_NFB4EOF_ERR[1]	New Frame before end-of-frame error indication of Channel interrupt. This bit is the status bit of New Frame before end-of-frame error interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.7 IPU Interrupt Status Register 7(IPU_INT_STAT_7)

This register contains part of IPUv3EX interrupts status bits. The status bits of the End-of-Scroll indication (EOS) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000218 (IPU_INT_STAT_7) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_EOS_31	0	IDMAC_EOS_29	IDMAC_EOS_28	IDMAC_EOS_27	0	0	IDMAC_EOS_24	IDMAC_EOS_23	0	0	0	0	0	0	0
W	w1c		w1c	w1c	w1c			w1c	w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-67. IPU Interrupt Status Register 7(IPU_INT_STAT_7)

Table 42-64. IPU_INT_STAT_7 Field Descriptions

Field	Description
31–0 *_EOS[1]	End of Scroll indication of Channel interrupt. This bit is the status bit of End of Scroll interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.8 IPU Interrupt Status Register 8 (IPU_INT_STAT_8)

This register contains part of IPUv3EX interrupts status bits. All the status bits of the End of Scroll indication (EOS) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE00021C (IPU_INT_STAT_8) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOS_52	IDMAC_EOS_51	0	0	0
W												w1c	w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOS_44	IDMAC_EOS_43	IDMAC_EOS_42	IDMAC_EOS_41	0	0	0	0	0	0	0	IDMAC_EOS_33	0
W				w1c	w1c	w1c	w1c								w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-68. IPU Interrupt Status Register 8 (IPU_INT_STAT_8)

Table 42-65. IPU_INT_STAT_8 Field Descriptions

Field	Description
31–0 *_EOS[1]	End of Scroll of Channel interrupt. This bit is the status bit of End Of Scroll interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.9 IPU Interrupt Status Register 9 (IPU_INT_STAT_9)

This register contains part of IPUv3EX interrupts status bits. This register holds the error interrupt indications coming from different modules within IPUv3EX. All the bits in this register are write one to clear.

Address 0xBASE+0xE000220 (IPU_INT_STAT_9) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI1_PUPE	CSI0_PUPE	ISP_PUPE	IC_VF_BUF_OVF	IC_ENC_BUF_OVF	IC_BAYER_BUF_OVF	0	0	0	0	0	0	0	0	0	0
W	w1c	w1c	w1c	w1c	w1c	w1c										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	VDI_FIFO1_OVF
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-69. IPU Interrupt Status Register 9 (IPU_INT_STAT_9)

Table 42-66. IPU_INT_STAT_9 Field Descriptions

Field	Description
31 CSI1_PUPE	CSI1_PUPE - CSI1 parameters update error interrupt. This bit indicates on an interrupt that is a result of an error generated by the CSI1. The error is generated in case where new frame arrived from the CSI1 before the completion of the CSI1's parameters update by the SRM 0 Interrupt is cleared. 1 Interrupt is requested.
30 CSI0_PUPE	CSI0_PUPE - CSI0 parameters update error interrupt. This bit indicates on an interrupt that is a result of an error generated by the CSI0. The error is generated in case where new frame arrived from the CSI0 before the completion of the CSI0's parameters update by the SRM 0 Interrupt is cleared. 1 Interrupt is requested.
28 IC_VF_BUF_OVF	This bit indicates on an interrupt that is a result of the IC Buffer overflow for view finder coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is cleared. 1 Interrupt is requested.

Table 42-66. IPU_INT_STAT_9 Field Descriptions

Field	Description
27 IC_ENC_BUF_OVF	This bit indicates on an interrupt that is a result of the IC Buffer overflow for encoding coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is cleared. 1 Interrupt is requested.
26 IC_BAYER_BUF_OVF	This bit indicates on an interrupt that is a result of the IC Buffer overflow for Bayer coming from the IC. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is cleared. 1 Interrupt is requested.
25-1	Reserved
0 VDI_FIFO1_OVF	FIFO1 overflow Interrupt1 The VDI generate FIFO1 overflow interrupt1 when write pointer of FIFO1 overrun read pointer. 0 Interrupt is cleared. 1 Interrupt is requested.

42.2.3.2.10 IPU Interrupt Status Register 10 (IPU_INT_STAT_10)

This register contains part of IPUv3EX interrupts status bits. This register holds error interrupt indications coming from different modules within IPUv3EX. All the bits in this register are write one to clear.

Address 0xBASE+0xE000224 (IPU_INT_STAT_10) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	ISP_RAM_HIST_OF	ISP_RAM_ST_OF	SMFC3_FRM_LOST	SMFC2_FRM_LOST	SMFC1_FRM_LOST	SMFC0_FRM_LOST
W											w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

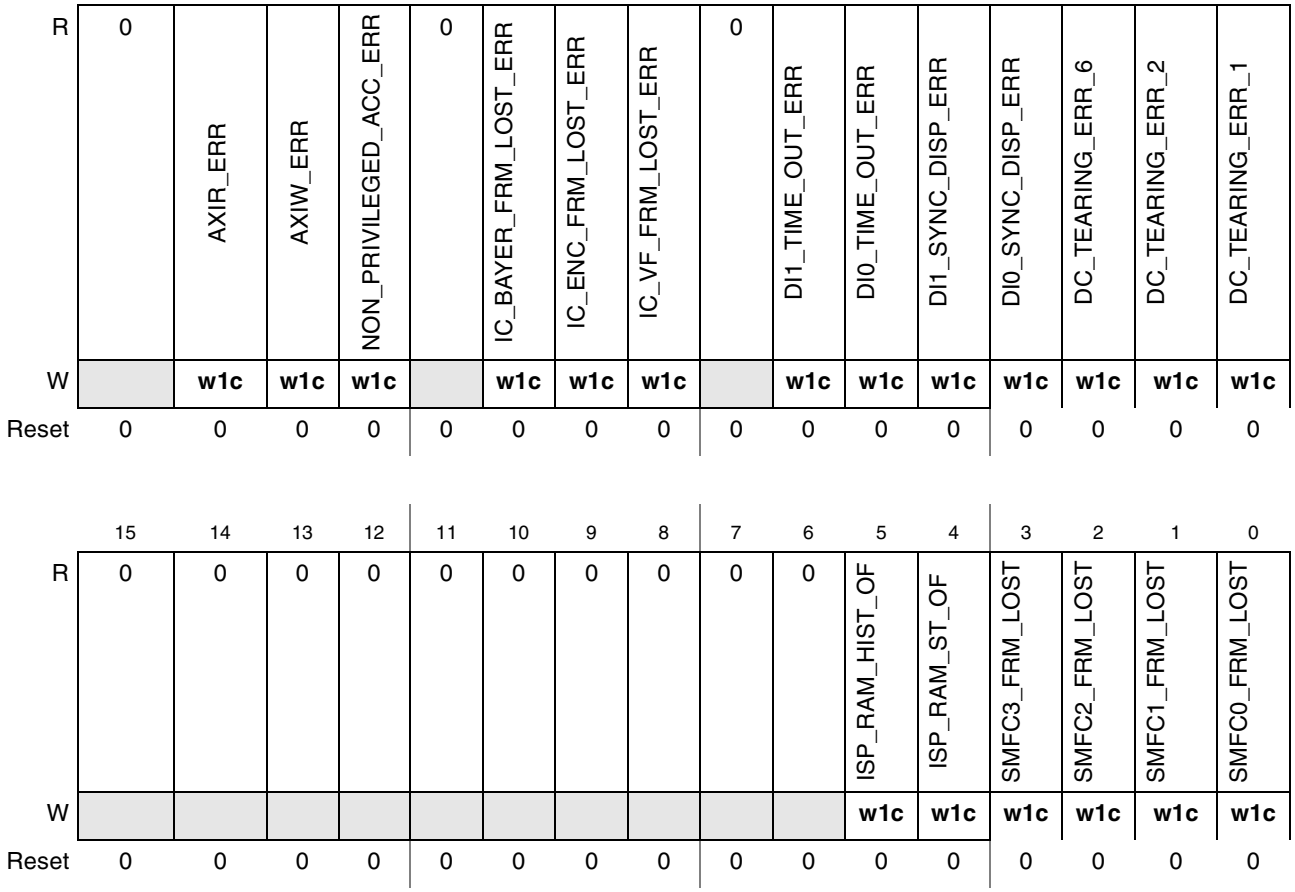


Figure 42-70. IPU Interrupt Status Register 10 (IPU_INT_STAT_10)

Table 42-67. IPU_INT_STAT_10 Field Descriptions

Field	Description
31	Reserved
30 AXIR_ERR	This bit indicates on an interrupt that is a result of AXI read access resulted with error response. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is cleared. 1 Interrupt is requested.
29 AXIW_ERR	This bit indicates on an interrupt that is a result of AXI write access resulted with error response. The user needs to write 1 to this bit in order to clear it. 0 Interrupt is cleared. 1 Interrupt is requested.
28 NON_PRIVILEGED_ACC_ERR	Non Privileged Access Error interrupt. This bit indicates on an interrupt that is a result of access the CPMEM or the DP memory in user mode 0 Interrupt is cleared. 1 Interrupt is requested.
27	Reserved

Table 42-67. IPU_INT_STAT_10 Field Descriptions

Field	Description
26 IC_BAYER_FRM_LOST_ERR	This bit indicates on an interrupt that is a result of IC's Bayer frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
25 IC_ENC_FRM_LOST_ERR	This bit indicates on an interrupt that is a result of IC's encoding frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
24 IC_VF_FRM_LOST_ERR	This bit indicates on an interrupt that is a result of IC's view finder frame lost. 0 Interrupt is disabled. 1 Interrupt is enabled.
23	Reserved
22 DI1_TIME_OUT_ERR	DI1 time out error interrupt This bit indicates on the interrupt that is a result of a time out error during a read access via DI1
21 DI0_TIME_OUT_ERR	DI0 time out error interrupt This bit indicates on the interrupt that is a result of a time out error during a read access via DI0
20 DI1_SYNC_DISP_ERR	DI1 Synchronous display error interrupt This bit indicates on the interrupt that is a result of an error during access to a synchronous display via DI1
19 DI0_SYNC_DISP_ERR	DI0 Synchronous display error interrupt This bit indicates on the interrupt that is a result of an error during access to a synchronous display via DI0
18 DC_TEARING_ERR_6	Tearing Error #6 interrupt This bit indicates on the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 6
17 DC_TEARING_ERR_2	Tearing Error #2 interrupt This bit indicates on the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 2
16 DC_TEARING_ERR_1	Tearing Error #1 interrupt This bit indicates on the interrupt that is a result of tearing error while the anti tearing mechanism is activated for DC channel 1
15–6	Reserved
5 ISP_RAM_HIST_OF	Histogram RAM overflow/underun Interrupt This bit indicates on an interrupt that is a result of over flow or under run in the Histogram FIFO 0 Interrupt is cleared. 1 Interrupt is requested.
4 ISP_RAM_ST_OF	Statistics RAM overflow/underun Interrupt This bit enables the interrupt that is a result of over flow or under run in the statistics FIFO 0 Interrupt is cleared. 1 Interrupt is requested.

Table 42-67. IPU_INT_STAT_10 Field Descriptions

Field	Description
3 SMFC3_FRM_LOST	Frame Lost of SMFC channel 3 interrupt. This bit indicates on an interrupt that is a result of a result of a Frame Lost of SMFC channel 3 0 Interrupt is cleared. 1 Interrupt is requested.
2 SMFC2_FRM_LOST	Frame Lost of SMFC channel 2 interrupt. This bit indicates on an interrupt that is a result of a result of a Frame Lost of SMFC channel 2 0 Interrupt is cleared. 1 Interrupt is requested.
1 SMFC1_FRM_LOST	Frame Lost of SMFC channel 1 interrupt. This bit indicates on an interrupt that is a result of a result of a Frame Lost of SMFC channel 1 0 Interrupt is cleared. 1 Interrupt is requested.
0 SMFC0_FRM_LOST	Frame Lost of SMFC channel 0 interrupt. This bit indicates on an interrupt that is a result of a result of a Frame Lost of SMFC channel 0 0 Interrupt is cleared. 1 Interrupt is requested.

42.2.3.2.11 IPU Interrupt Status Register 11 (IPU_INT_STAT_11)

This register contains part of IPUv3EX interrupts status bits. The status bits of the end-of-band indication (EOBND) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000228 (IPU_INT_STAT_11) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_EOBND_12	IDMAC_EOBND_11	0	0	0	0	0	IDMAC_EOBND_5	0	IDMAC_EOBND_3	IDMAC_EOBND_2	IDMAC_EOBND_1	IDMAC_EOBND_0
W				w1c	w1c						w1c		w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

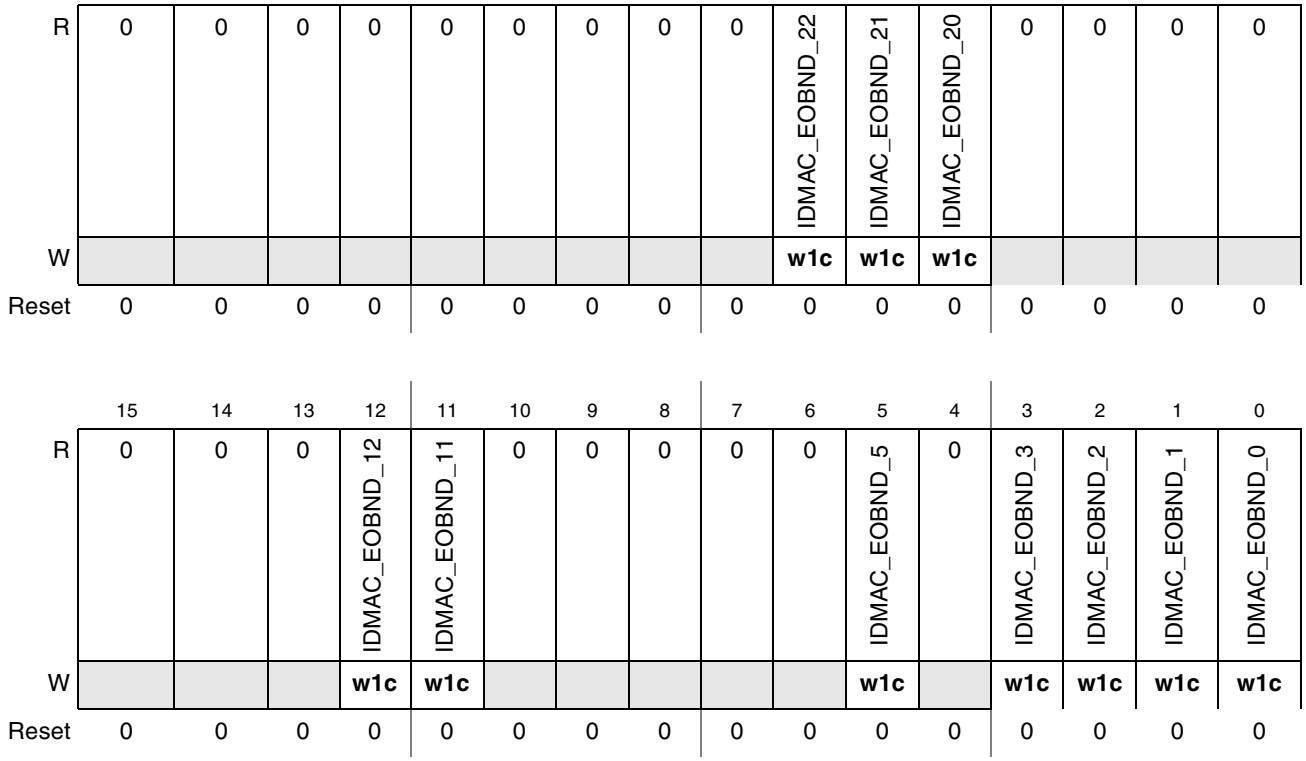


Figure 42-71. IPU Interrupt Status Register 11 (IPU_INT_STAT_11)

Table 42-68. IPU_INT_STAT_11 Field Descriptions

Field	Description
31–0 *_EOBND[1]	end-of-band indication of Channel interrupt. This bit is the status bit of end-of-band interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.12 IPU Interrupt Status Register 12 (IPU_INT_STAT_12)

This register contains part of IPUv3EX interrupts status bits. The status bits of the end-of-band indication (EOBND) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE00022C (IPU_INT_STAT_12) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_EOBND_50	IDMAC_EOBND_49	IDMAC_EOBND_48
W														w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_EOBND_47	IDMAC_EOBND_46	IDMAC_EOBND_45	0	0	0	0	0	0	0	0	0	0	0	0	0
W	w1c	w1c	w1c													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-72. IPU Interrupt Status Register 12 (IPU_INT_STAT_12)

Table 42-69. IPU_INT_STAT_12 Field Descriptions

Field	Description
31–0 *_EOBND[1]	end-of-band indication of Channel interrupt. This bit is the status bit of end-of-band interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

1 Indicates the corresponding DMA channel number.

42.2.3.2.13 IPU Interrupt Status Register 13 (IPU_INT_STAT_13)

This register contains part of IPUv3EX interrupts status bits. The status bits of the Threshold crossing indication (TH) of DMA Channels interrupts [31:0] can be found in this register.

Address 0xBASE+0xE000230 (IPU_INT_STAT_13) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_TH_31	0	IDMAC_TH_29	IDMAC_TH_28	IDMAC_TH_27	0	0	IDMAC_TH_24	IDMAC_TH_23	IDMAC_TH_22	IDMAC_TH_21	IDMAC_TH_20	0	IDMAC_TH_18	IDMAC_TH_17	0
W	w1c		w1c	w1c	w1c			w1c	w1c	w1c	w1c	w1c		w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_TH_15	IDMAC_TH_14	IDMAC_TH_13	IDMAC_TH_12	IDMAC_TH_11	IDMAC_TH_10	IDMAC_TH_9	IDMAC_TH_8	IDMAC_TH_7	IDMAC_TH_6	IDMAC_TH_5	IDMAC_TH_4	IDMAC_TH_3	IDMAC_TH_2	IDMAC_TH_1	IDMAC_TH_0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-73. IPU Interrupt Status Register 13 (IPU_INT_STAT_13)

Table 42-70. IPU_INT_STAT_13 Field Descriptions

Field	Description
31–0 *_TH[1]	Threshold crossing indication of Channel interrupt. This bit is the status bit of Threshold crossing interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.14 IPU Interrupt Status Register 14 (IPU_INT_STAT_14)

This register contains part of IPUv3EX interrupts status bits. The status bits of the Threshold crossing indication (TH) of DMA Channels interrupts [63:32] can be found in this register.

Address 0xBASE+0xE000234 (IPU_INT_STAT_14) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_TH_52	IDMAC_TH_51	IDMAC_TH_50	IDMAC_TH_49	IDMAC_TH_48
W												w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_TH_47	IDMAC_TH_46	IDMAC_TH_45	IDMAC_TH_44	IDMAC_TH_43	IDMAC_TH_42	IDMAC_TH_41	IDMAC_TH_40	0	0	0	0	0	0	IDMAC_TH_33	0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c							w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-74. IPU Interrupt Status Register 14 (IPU_INT_STAT_14)

Table 42-71. IPU_INT_STAT_14 Field Descriptions

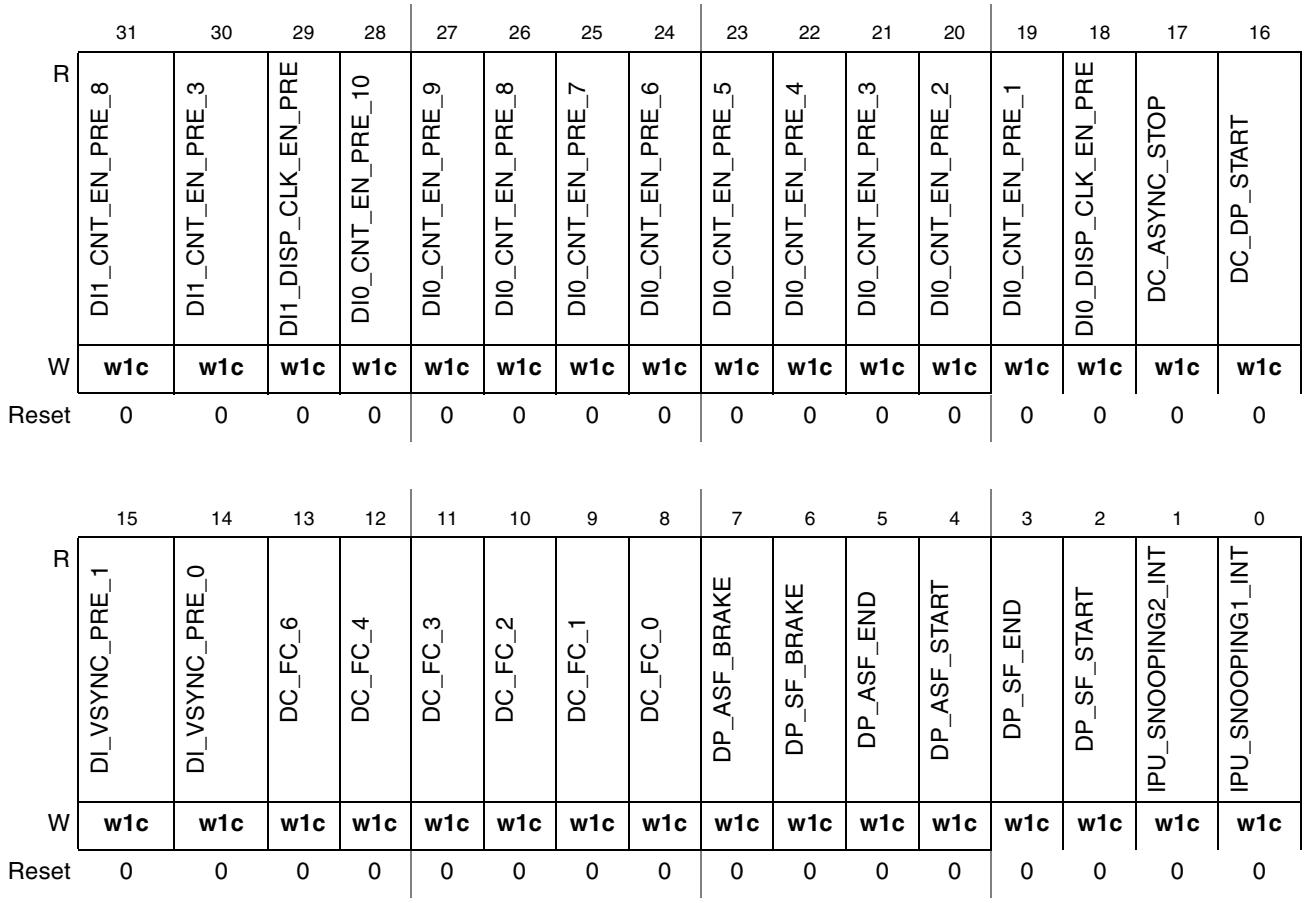
Field	Description
31–0 *_TH[1]	Threshold crossing indication of Channel interrupt. This bit is the status bit of Threshold crossing interrupt of Channel #n. 0 Interrupt is cleared. 1 Interrupt is requested.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.15 IPU Interrupt Status Register 15 (IPU_INT_STAT_15)

This register contains part of IPUv3EX interrupts status bits. The status bits of general purpose interrupts can be found in this register.

Address 0xBASE+0xE000238 (IPU_INT_STAT_15) Access: User read-write



	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DI1_CNT_EN_PRE_8	DI1_CNT_EN_PRE_3	DI1_DISP_CLK_EN_PRE	DI0_CNT_EN_PRE_10	DI0_CNT_EN_PRE_9	DI0_CNT_EN_PRE_8	DI0_CNT_EN_PRE_7	DI0_CNT_EN_PRE_6	DI0_CNT_EN_PRE_5	DI0_CNT_EN_PRE_4	DI0_CNT_EN_PRE_3	DI0_CNT_EN_PRE_2	DI0_CNT_EN_PRE_1	DI0_DISP_CLK_EN_PRE	DC_ASYNC_STOP	DC_DP_START
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI_VSYNC_PRE_1	DI_VSYNC_PRE_0	DC_FC_6	DC_FC_4	DC_FC_3	DC_FC_2	DC_FC_1	DC_FC_0	DP_ASF_BRAKE	DP_SF_BRAKE	DP_ASF_END	DP_ASF_START	DP_SF_END	DP_SF_START	IPU_SNOOPING2_INT	IPU_SNOOPING1_INT
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-75. IPU Interrupt Status Register 15 (IPU_INT_STAT_15)

Table 42-72. IPU_INT_STAT_15 Field Descriptions

Field	Description
31 DI1_CNT_EN_PRE_8	This bit indicates on the interrupt that is a result of a trigger generated by counter #8 of DI1 0 Interrupt is cleared. 1 Interrupt is requested.
30 DI1_CNT_EN_PRE_3	This bit indicates on the interrupt that is a result of a trigger generated by counter #3 of DI1 0 Interrupt is cleared. 1 Interrupt is requested.
29 DI1_DISP_CLK_EN_PRE	0 Interrupt is cleared. 1 Interrupt is requested.
28 DI0_CNT_EN_PRE_10	This bit indicates on the interrupt that is a result of a trigger generated by counter #10 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.

Table 42-72. IPU_INT_STAT_15 Field Descriptions (continued)

Field	Description
27 DI0_CNT_EN_PRE_9	This bit indicates on the interrupt that is a result of a trigger generated by counter #9 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
26 DI0_CNT_EN_PRE_8	This bit indicates on the interrupt that is a result of a trigger generated by counter #8 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
25 DI0_CNT_EN_PRE_7	This bit indicates on the interrupt that is a result of a trigger generated by counter #7 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
24 DI0_CNT_EN_PRE_6	This bit indicates on the interrupt that is a result of a trigger generated by counter #6 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
23 DI0_CNT_EN_PRE_5	This bit indicates on the interrupt that is a result of a trigger generated by counter #5 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
22 DI0_CNT_EN_PRE_4	This bit indicates on the interrupt that is a result of a trigger generated by counter #4 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
21 DI0_CNT_EN_PRE_3	This bit indicates on the interrupt that is a result of a trigger generated by counter #3 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
20 DI0_CNT_EN_PRE_2	This bit indicates on the interrupt that is a result of a trigger generated by counter #2 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
19 DI0_CNT_EN_PRE_1	This bit indicates on the interrupt that is a result of a trigger generated by counter #1 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
18 DI0_CNT_EN_PRE_0	This bit indicates on the interrupt that is a result of a trigger generated by counter #0 of DI0 0 Interrupt is cleared. 1 Interrupt is requested.
17 DC_ASYNC_STOP	This bit indicates on an interrupt asserted anytime the DP stops an async flow and moves to a sync flow 0 Interrupt is cleared. 1 Interrupt is requested.

Table 42-72. IPU_INT_STAT_15 Field Descriptions (continued)

Field	Description
16 DC_DP_START	This bit indicates on an interrupt asserted anytime the DP start a new sync or async flow or when an async flow is interrupted by a sync flow 0 Interrupt is cleared. 1 Interrupt is requested.
15 DI_VSYNC_PRE_1	DI1 interrupt indicating of a VSYNC signal asserted 2 rows before the VSYNC sent to the display 0 Interrupt is cleared. 1 Interrupt is requested.
14 DI_VSYNC_PRE_0	DI0 interrupt indicating of a VSYNC signal asserted 2 rows before the VSYNC sent to the display 0 Interrupt is cleared. 1 Interrupt is requested.
13 DC_FC_6	DC Frame Complete on channel #6 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
12 DC_FC_4	DC Frame Complete on channel #4 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
11 DC_FC_3	DC Frame Complete on channel #3 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
10 DC_FC_2	DC Frame Complete on channel #2 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
9 DC_FC_1	DC Frame Complete on channel #1 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
8 DC_FC_0	DC Frame Complete on channel #0 interrupt indication 0 Interrupt is cleared. 1 Interrupt is requested.
7 DP_ASF_BRAKE	DP Async Flow Brake indication interrupt. This bit indicates on the interrupt that is a result of the async flow brake at the DP 0 Interrupt is cleared. 1 Interrupt is requested.
6 DP_SF_BRAKE	DP Sync Flow Brake indication interrupt. This bit indicates on the interrupt that is a result of the Sync flow brake at the DP 0 Interrupt is cleared. 1 Interrupt is requested.
5 DP_SF_END	DP Async Flow End indication interrupt. This bit indicates on the interrupt that is a result of the Async flow end at the DP 0 Interrupt is cleared. 1 Interrupt is requested.
4 DP_ASF_START	DP Async Flow Start indication interrupt. This bit indicates on the interrupt that is a result of the Async flow start at the DP 0 Interrupt is cleared. 1 Interrupt is requested.

Table 42-72. IPU_INT_STAT_15 Field Descriptions (continued)

Field	Description
3 DP_SF_END	DP Sync Flow End indication interrupt. This bit indicates on the interrupt that is a result of the Sync flow end at the DP 0 Interrupt is cleared. 1 Interrupt is requested.
2 DP_SF_START	DP Sync Flow Start indication interrupt. This bit indicates on the interrupt that is a result of the Sync flow start at the DP 0 Interrupt is cleared. 1 Interrupt is requested.
1 IPU_SNOOPING2_INT	IPUv3EX snooping 2 event indication interrupt. This bit indicates on the interrupt that is a result of the detection of a snooping 2 signal assertion coming to the IPUv3EX. 0 Interrupt is cleared. 1 Interrupt is requested.
0 IPU_SNOOPING1_INT	IPUv3EX snooping 1 event indication interrupt. This bit indicates on the interrupt that is a result of the detection of a snooping 1 signal assertion coming to the IPUv3EX. 0 Interrupt is cleared. 1 Interrupt is requested.

42.2.3.2.16 IPU Current Buffer Register 0 (IPU_CUR_BUF_0)

This register contains the current buffer status information bit for each DMA channel.

The register is shown in Figure 42-76, and the register fields are described in Table 42-73.

Address 0xBASE+0xE00023C (IPU_CUR_BUF_0) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_CUR_BUF_15	DMA_CH_CUR_BUF_14	0	DMA_CH_CUR_BUF_12	DMA_CH_CUR_BUF_11	0	0	0	DMA_CH_CUR_BUF_7	DMA_CH_CUR_BUF_6	DMA_CH_CUR_BUF_5	DMA_CH_CUR_BUF_4	DMA_CH_CUR_BUF_3	DMA_CH_CUR_BUF_2	DMA_CH_CUR_BUF_1	DMA_CH_CUR_BUF_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

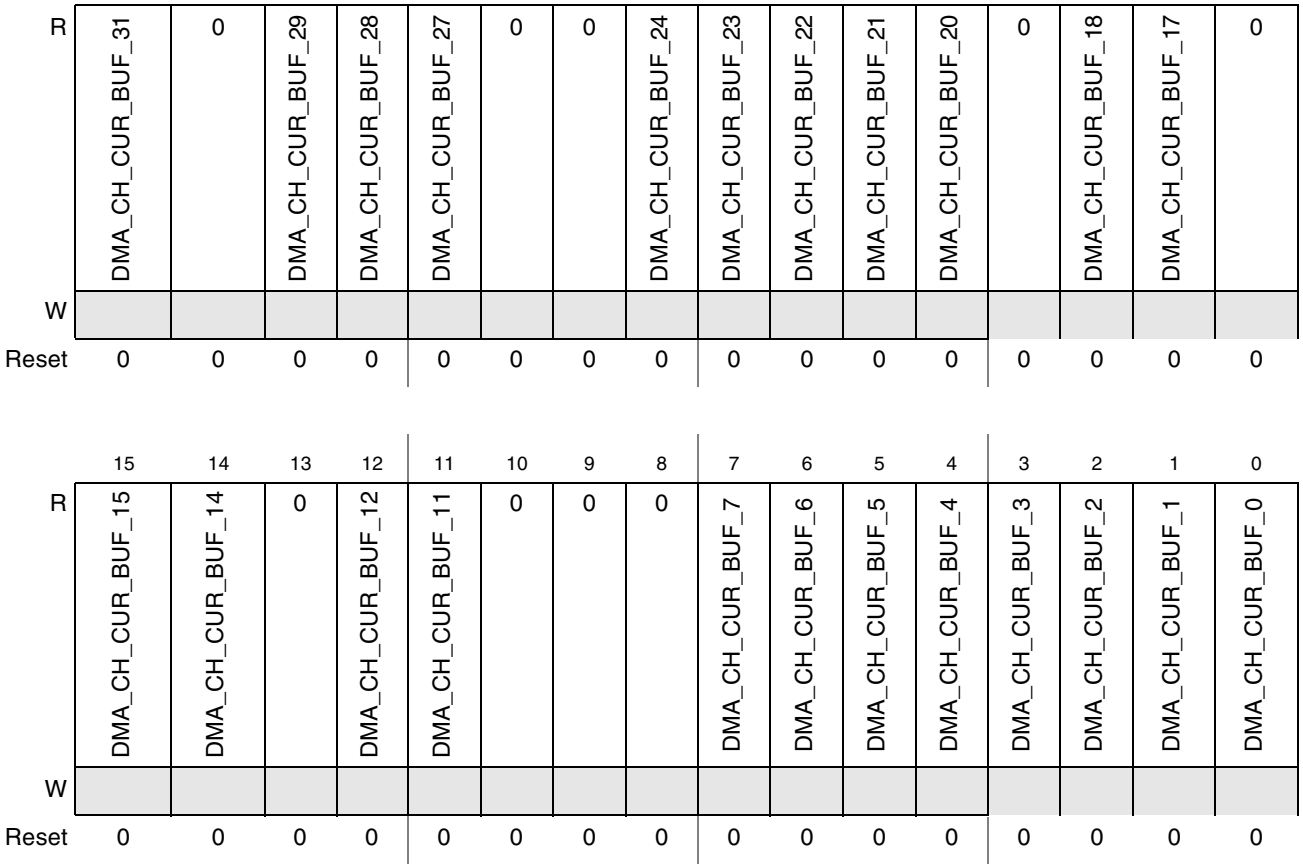


Figure 42-76. IPU Current Buffer Register 0 (IPU_CUR_BUF_0)

Table 42-73. IPU_CUR_BUF_0 Field Descriptions

Field	Description
31–0 DMA_CH_CUR_BUF_*[1]	Current buffer. This bit indicates which buffer is in use by DMA when double buffer mode is selected. 0 Current buffer used by DMA is buffer 0. 1 Current buffer used by DMA is buffer 1.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.17 IPU Current Buffer Register 1 (IPU_CUR_BUF_1)

This register contains the current buffer status information bit for each DMA channel.

The register is shown in Figure 42-77, and the register fields are described in Table 42-74.

Address 0xBASE+0xE000240 (IPU_CUR_BUF_1) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_CUR_BUF_52	DMA_CH_CUR_BUF_51	DMA_CH_CUR_BUF_50	DMA_CH_CUR_BUF_49	DMA_CH_CUR_BUF_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_CUR_BUF_47	DMA_CH_CUR_BUF_46	DMA_CH_CUR_BUF_45	DMA_CH_CUR_BUF_44	DMA_CH_CUR_BUF_43	DMA_CH_CUR_BUF_42	DMA_CH_CUR_BUF_41	DMA_CH_CUR_BUF_40	0	0	0	0	0	0	DMA_CH_CUR_BUF_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-77. IPU Current Buffer Register 1 (IPU_CUR_BUF_1)

Table 42-74. IPU_CUR_BUF_1 Field Descriptions

Field	Description
31–0 DMA_CH_CUR_BUF_*[1]	Current buffer. This bit indicates which buffer is in use by DMA when double buffer mode is selected. 0 Current buffer used by DMA is buffer 0. 1 Current buffer used by DMA is buffer 1.

1 Indicates the corresponding DMA channel number.

42.2.3.2.18 IPU Alternate Current Buffer Register 0 (IPU_ALT_CUR_BUF_0)

This register contains the current buffer status information bit for each DMA channel.

The register is shown in Figure 42-78, and the register fields are described in Table 42-75.

Address 0xBASE+0xE000244 (IPU_ALT_CUR_BUF_0) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	DMA_CH_ALT_CUR_BUF_29	0	0	0	0	DMA_CH_ALT_CUR_BUF_24	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DMA_CH_ALT_CUR_BUF_7	DMA_CH_ALT_CUR_BUF_6	DMA_CH_ALT_CUR_BUF_5	DMA_CH_ALT_CUR_BUF_4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-78. IPU Alternate Current Buffer Register 0 (IPU_ALT_CUR_BUF_0)

Table 42-75. IPU_ALT_CUR_BUF_0 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_CUR_BUF_*[1]	Current buffer. This bit indicates which buffer is in use by DMA when double buffer mode is selected. 0 Current buffer used by DMA is buffer 0. 1 Current buffer used by DMA is buffer 1.

1 Indicates the corresponding DMA channel number.

42.2.3.2.19 IPU Alternate Current Buffer Register 1 (IPU_ALT_CUR_BUF_1)

This register contains the current buffer status information bit for each DMA channel. The register is shown in Figure 42-79, and the register fields are described in Table 42-76.

Address 0xBASE+0xE000248 (IPU_ALT_CUR_BUF_1) Access: User read-only

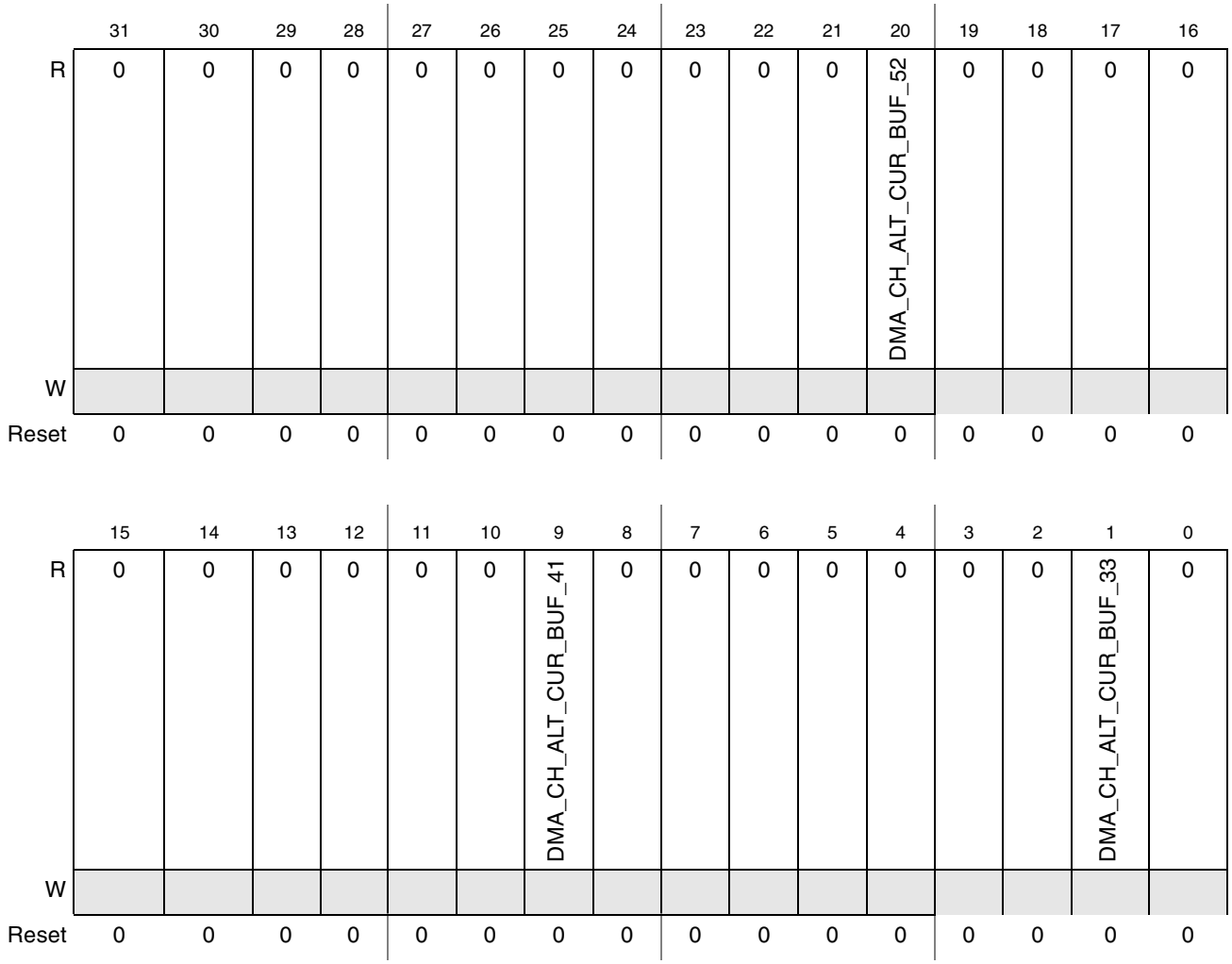


Figure 42-79. IPU Alternate Current Buffer Register 1 (IPU_ALT_CUR_BUF_1)

Table 42-76. IPU_ALT_CUR_BUF_1 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_CUR_BUF_*[1]	Current buffer. This bit indicates which buffer is in use by DMA when double buffer mode is selected. 0 Current buffer used by DMA is buffer 0. 1 Current buffer used by DMA is buffer 1.

1 Indicates the corresponding DMA channel number.

42.2.3.2.20 IPU Shadow Registers Memory Status Register (IPU_SRM_STAT)

The register contains status bits of SRM updates. There is a bit for each module. the bit is set when the SRM is currently updating the module's registers. When the SRM completes updating the registers of the

module the bit is cleared. SW should not update the module's registers while it is being updated by the SRM.

The register is shown in Figure 42-80, and the register fields are described in Table 42-77.

Address 0xBASE+0xE00024C (IPU_SRM_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DI1_SRM_STAT	DI0_SRM_STAT	CSI1_SRM_STAT	CSI0_SRM_STAT	DC_6_SRM_STAT	DC_2_SRM_STAT	ISP_SRM_STAT	DP_A1_SRM_STAT	DP_A0_SRM_STAT	DP_S_SRM_STAT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-80. IPU Shadow Registers Memory Status Register (IPU_SRM_STAT)

Table 42-77. IPU_SRM_STAT Field Descriptions

Field	Description
31-6	Reserved
5 DI1_SRM_STAT	DI1 SRM STAT This bit indicates that the SRM is currently updating the DI1 registers 1 SRM is busy updating the DI1 registers 0 SRM is not updating the DI1 registers
5 DI0_SRM_STAT	DI0 SRM STAT This bit indicates that the SRM is currently updating the DI0 registers 1 SRM is busy updating the DI0 registers 0 SRM is not updating the DI0 registers
4	
3	

Table 42-77. IPU_SRM_STAT Field Descriptions (continued)

Field	Description
2 DC_6_SRM_STAT	DC group #6 SRM STAT This bit indicates that the SRM is currently updating the DC group #6 registers 1 SRM is busy updating the DC registers 0 SRM is not updating the DC registers
2 DC_2_SRM_STAT	DC group #2 SRM STAT This bit indicates that the SRM is currently updating the DC group #2 registers 1 SRM is busy updating the DC group #6 registers 0 SRM is not updating the DC group #2 registers
1	
0 DP_A1_SRM_STAT	DP ASYNC1 FLOW SRM STAT This bit indicates that the SRM is currently updating the DP async flow 1 registers 1 SRM is busy updating the DP sync flow registers 0 SRM is not updating the DP sync flow registers
0 DP_A0_SRM_STAT	DP ASYNC0 FLOW SRM STAT This bit indicates that the SRM is currently updating the DP async flow 0 registers 1 SRM is busy updating the DP async flow 0 registers 0 SRM is not updating the DP async flow 0 registers
0 DP_S_SRM_STAT	DP SYNC FLOW SRM STAT This bit indicates that the SRM is currently updating the DP sync flow registers 1 SRM is busy updating the DP sync flow registers 0 SRM is not updating the DP sync flow registers

42.2.3.2.21 IPU Processing Tasks Status Register (IPU_PROC_TASKS_STAT)

This register contains status bits for IPUv3EX's tasks.

The register is shown in Figure 42-81, and the register fields are described in Table 42-78.

Address 0xBASE+0xE000250 (IPU_PROC_TASKS_STAT) Access: User read-only

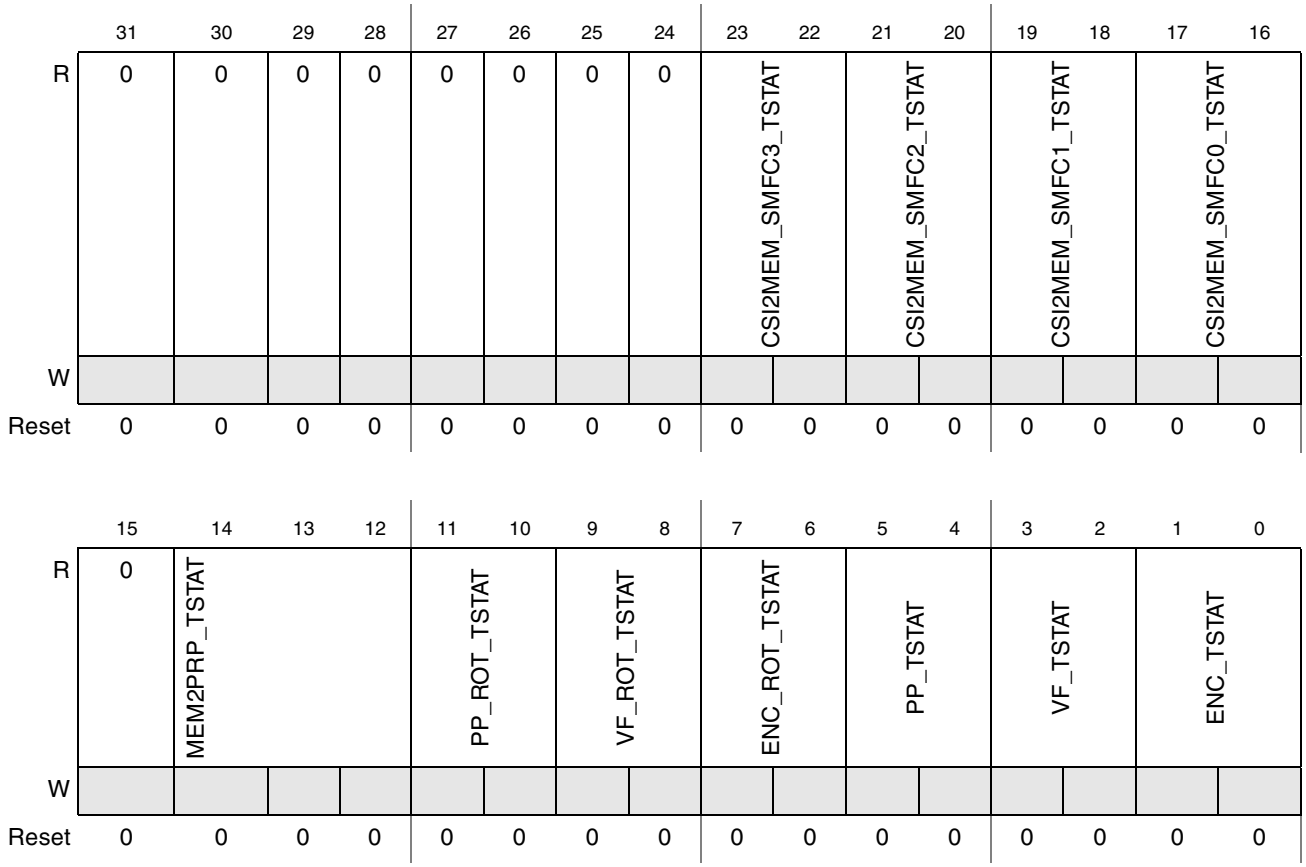


Figure 42-81. IPU Processing Tasks Status Register (IPU_PROC_TASKS_STAT)

Table 42-78. IPU_PROC_TASKS_STAT Field Descriptions

Field	Description
31-24	Reserved
23-22 CSI2MEM_SMFC3_TSTAT	Status of the SMFC flow #3 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
21-20 CSI2MEM_SMFC2_TSTAT	Status of the SMFC flow #2 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
19-18 CSI2MEM_SMFC1_TSTAT	Status of the SMFC flow #1 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready

Table 42-78. IPU_PROC_TASKS_STAT Field Descriptions (continued)

Field	Description
17-16 CSI2MEM_SMFC0_TSTAT	Status of the SMFC flow #0 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
15	Reserved
14-12 MEM2PRP_TSTAT	Status of the pre processing tasks (viewfinder and encoding) when the source is coming from the memory. 3'b000 IDLE - Both pre processing tasks are idle 3'b001 BOTH_ACTIVE - Both pre processing tasks are idle 3'b010 ENC_ACTIVE - Encoding task is active 3'b011 VF_ACTIVE - View finder task is active 3'b100 BOTH_PAUSE - both tasks are paused 3'b101 Reserved 3'b110 Reserved 3'b111 Reserve
11-10 PP_ROT_TSTAT	Status of the rotation for post processing task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
9-8 VF_ROT_TSTAT	Status of the rotation for viewfinder task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
7-6 ENC_ROT_TSTAT	Status of the rotation for encoding task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
5-4 PP_TSTAT	Status of the post processing task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
3-2 VF_TSTAT	Status of the viewfinder task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
1-0 ENC_TSTAT	Status of the encoding task 2'b00 IDLE - The task is idle 2'b01 ACTIVE - The primary flow of this task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready

42.2.3.2.22 IPU Display Tasks Status Register (IPU_DISP_TASKS_STAT)

This register contains status bits for IPUv3EX's tasks.

The register is shown in Figure 42-82, and the register fields are described in Table 42-79.

Address 0xBASE+0xE000254 (IPU_DISP_TASKS_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	DC_ASYNC2_CUR_FLOW	DC_ASYNC2_STAT			0	0	DC_ASYNC1_STAT		DP_ASYNC_CUR_FLOW	DP_ASYNC_STAT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-82. IPU Display Tasks Status Register (IPU_DISP_TASKS_STAT)

Table 42-79. IPU_DISP_TASKS_STAT Field Descriptions

Field	Description
31-12	Reserved
11 DC_ASYNC2_CUR_FLOW	Current asynchronous #2 flow via the DC 1- alternate flow 0 - main flow
10-8 DC_ASYNCH2_STAT	Status of the Asynchronous flow #2 through the DC 3'b000 IDLE - the task is idle 3'b001 PRIM_ACTIVE - The primary flow of this task is currently active 3'b010 ALT_ACTIVE - The alternate flow of this task is currently active 3'b011 UPDATE_PARAM - The FSU is busy updating parameters from the SRM 3'b100 PAUSE - The task is paused 3'b101 Reserved 3'b110 Reserved 3'b111 Reserved
7-6	Reserved

Table 42-79. IPU_DISP_TASKS_STAT Field Descriptions (continued)

Field	Description
5-4 DC_ASYNC1_STAT	Status of the Asynchronous flow #1 through the DC (ch 28) 2'b00 IDLE - The task is idle 2'b01 ACTIVE - This task is currently active 2'b10 WAIT_FOR_ READY - The task is waiting for a buffer to be ready
5-3	Reserved
3 DP_ASYNC_CUR_FLOW	Current asynchronous flow via the DP 1- alternate flow 0 - main flow
2-0 DP_ASYNC_STAT	Status of the Asynchronous flow through the DP 3'b000 IDLE - the task is idle 3'b001 PRIM_ACTIVE - The primary flow of this task is currently active 3'b010 ALT_ACTIVE - The alternate flow of this task is currently active 3'b011 UPDATE_PARAM - The FSU is busy updating parameters from the SRM 3'b100 PAUSE - The task is paused 3'b101 Reserved 3'b110 Reserved 3'b111 Reserved

42.2.3.2.23 IPU Triple Current Buffer Register 0 (IPU_TRIPLE_CUR_BUF_0)

This register contains the current buffer status information for triple buffer mode for each DMA channel. The register is shown in Figure 42-83, and the register fields are described in Table 42-80.

Address 0xBASE+0xE000258 (IPU_TRIPLE_CUR_BUF_0) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	DMA_CH_TRIPLE_CUR_BUF_13		0	0	0	0	DMA_CH_TRIPLE_CUR_BUF_10		DMA_CH_TRIPLE_CUR_BUF_9		DMA_CH_TRIPLE_CUR_BUF_8	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-83. IPU Triple Current Buffer Register 0 (IPU_TRIPLE_CUR_BUF_0)

Table 42-80. IPU_TRIPLE_CUR_BUF_0 Field Descriptions

Field	Description
31–0 DMA_CH_TRIPLE_CUR_BUF_*	Current buffer for triple buffer mode. This bits indicate which buffer is in use by the DMA when triple buffer mode is selected. Each pair of bits indicate the corresponding DMA channel (bits [1:0] correspond to ch #0; (bits [3:2] correspond to ch #1, etc.) 00 Current buffer used by DMA is buffer 0. 01 Current buffer used by DMA is buffer 1. 10 Current buffer used by DMA is buffer 2. 11. NA

42.2.3.2.24 IPU Triple Current Buffer Register 1 (IPU_TRIPLE_CUR_BUF_1)

This register contains the current buffer status information for triple buffer mode for each DMA channel.

The register is shown in Figure 42-84, and the register fields are described in Table 42-81.

Address 0xBASE+0xE00025C (IPU_TRIPLE_CUR_BUF_1) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	DMA_CH_TRIPLE_CUR_BUF_28		DMA_CH_TRIPLE_CUR_BUF_27		0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_TRIPLE_CUR_BUF_23		0	0	DMA_CH_TRIPLE_CUR_BUF_21		0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-84. IPU Triple Current Buffer Register 1 (IPU_TRIPLE_CUR_BUF_1)

Table 42-81. IPU_TRIPLE_CUR_BUF_1 Field Descriptions

Field	Description
31–0 DMA_CH_TRIPLE_CUR_BUF_*	Current buffer for triple buffer mode. This bits indicate which buffer is in use by the DMA when triple buffer mode is selected. Each pair of bits indicate the corresponding DMA channel (bits [1:0] correspond to ch #0; (bits [3:2] correspond to ch #1, etc.) 00 Current buffer used by DMA is buffer 0. 01 Current buffer used by DMA is buffer 1. 10 Current buffer used by DMA is buffer 2. 11. NA

42.2.3.2.25 IPU Triple Current Buffer Register 2 (IPU_TRIPLE_CUR_BUF_2)

This register contains the current buffer status information for triple buffer mode for each DMA channel. The register is shown in Figure 42-85, and the register fields are described in Table 42-82.

Address 0xBASE+0xE000260 (IPU_TRIPLE_CUR_BUF_2) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-85. IPU Triple Current Buffer Register 2 (IPU_TRIPLE_CUR_BUF_2)

Table 42-82. IPU_TRIPLE_CUR_BUF_2 Field Descriptions

Field	Description
31–0 DMA_CH_TRIPLE_CUR_BUF_*	Current buffer for triple buffer mode. This bits indicate which buffer is in use by DMA when triple buffer mode is selected. Each pair of bits indicate the corresponding DMA channel (bits [1:0] correspond to ch #0; (bits [3:2] correspond to ch #1, etc.) 00 Current buffer used by DMA is buffer 0. 01 Current buffer used by DMA is buffer 1. 10 Current buffer used by DMA is buffer 2. 11. NA

42.2.3.2.26 IPU Triple Current Buffer Register 3 (IPU_TRIPLE_CUR_BUF_3)

This register contains the current buffer status information for triple buffer mode for each DMA channel. The register is shown in Figure 42-86, and the register fields are described in Table 42-83.

Address 0xBASE+0xE000264 (IPU_TRIPLE_CUR_BUF_3) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-86. IPU Triple Current Buffer Register 3 (IPU_TRIPLE_CUR_BUF_3)

Table 42-83. IPU_TRIPLE_CUR_BUF_3 Field Descriptions

Field	Description
31–0 DMA_CH_TRIPLE_CUR_BUF_*	Current buffer for triple buffer mode. This bits indicate which buffer is in use by DMA when triple buffer mode is selected. Each pair of bits indicate the corresponding DMA channel (bits [1:0] correspond to ch #0; (bits [3:2] correspond to ch #1, etc.) 00 Current buffer used by DMA is buffer 0. 01 Current buffer used by DMA is buffer 1. 10 Current buffer used by DMA is buffer 2. 11. NA

42.2.3.2.27 IPU Channels Buffer 0 Ready 0 Register (IPU_CH_BUF0_RDY0)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (31-0). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF0_RDY0_CLR bit.

The register is shown in Figure 42-87, and the register fields are described in Table 42-84.

Address 0xBASE+0xE000268 (IPU_CH_BUF0_RDY0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_CH_BUF0_RDY_31	0	DMA_CH_BUF0_RDY_29	DMA_CH_BUF0_RDY_28	DMA_CH_BUF0_RDY_27	0	0	DMA_CH_BUF0_RDY_24	DMA_CH_BUF0_RDY_23	DMA_CH_BUF0_RDY_22	DMA_CH_BUF0_RDY_21	DMA_CH_BUF0_RDY_20	0	DMA_CH_BUF0_RDY_18	DMA_CH_BUF0_RDY_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_BUF0_RDY_15	DMA_CH_BUF0_RDY_14	DMA_CH_BUF0_RDY_13	DMA_CH_BUF0_RDY_12	DMA_CH_BUF0_RDY_11	DMA_CH_BUF0_RDY_10	DMA_CH_BUF0_RDY_9	DMA_CH_BUF0_RDY_8	DMA_CH_BUF0_RDY_7	DMA_CH_BUF0_RDY_6	DMA_CH_BUF0_RDY_5	DMA_CH_BUF0_RDY_4	DMA_CH_BUF0_RDY_3	DMA_CH_BUF0_RDY_2	DMA_CH_BUF0_RDY_1	DMA_CH_BUF0_RDY_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-87. IPU Channels Buffer 0 Ready 0 Register (IPU_CH_BUF0_RDY0)

Table 42-84. IPU_CH_BUF0_RDY0 Field Descriptions

Field	Description
31–0 DMA_CH_BUF0_RDY_*[1]	Buffer 0 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.28 IPU Channels Buffer 0 Ready 1 Register (IPU_CH_BUF0_RDY1)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (63-32). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF0_RDY1_CLR bit.

The register is shown in Figure 42-88, and the register fields are described in Table 42-85.

f

Address 0xBASE+0xE00026C (IPU_CH_BUF0_RDY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_BUF0_RDY_52	DMA_CH_BUF0_RDY_51	DMA_CH_BUF0_RDY_50	DMA_CH_BUF0_RDY_49	DMA_CH_BUF0_RDY_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_BUF0_RDY_47	DMA_CH_BUF0_RDY_46	DMA_CH_BUF0_RDY_45	DMA_CH_BUF0_RDY_44	DMA_CH_BUF0_RDY_43	DMA_CH_BUF0_RDY_42	DMA_CH_BUF0_RDY_41	DMA_CH_BUF0_RDY_40	0	0	0	0	0	0	DMA_CH_BUF0_RDY_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-88. IPU Channels Buffer 0 Ready 1 Register (IPU_CH_BUF0_RDY1)

Table 42-85. IPU_CH_BUF0_RDY1 Field Descriptions

Field	Description
31–0 DMA_CH_BUF0_RDY_*[1]	Buffer 0 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.29 IPU Channels Buffer 1 Ready 0 Register (IPU_CH_BUF1_RDY0)

The register contains buffer 1 ready control information for 32 IPUv3's DMA channels (31-0). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF1_RDY0_CLR bit.

The register is shown in Figure 42-89, and the register fields are described in Table 42-86.

Address 0xBASE+0xE000270 (IPU_CH_BUF1_RDY0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_CH_BUF1_RDY_31	0	DMA_CH_BUF1_RDY_29	DMA_CH_BUF1_RDY_28	DMA_CH_BUF1_RDY_27	0	0	DMA_CH_BUF1_RDY_24	DMA_CH_BUF1_RDY_23	DMA_CH_BUF1_RDY_22	DMA_CH_BUF1_RDY_21	DMA_CH_BUF1_RDY_20	0	DMA_CH_BUF1_RDY_18	DMA_CH_BUF1_RDY_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_BUF1_RDY_15	DMA_CH_BUF1_RDY_14	DMA_CH_BUF1_RDY_13	DMA_CH_BUF1_RDY_12	DMA_CH_BUF1_RDY_11	DMA_CH_BUF1_RDY_10	DMA_CH_BUF1_RDY_9	DMA_CH_BUF1_RDY_8	DMA_CH_BUF1_RDY_7	DMA_CH_BUF1_RDY_6	DMA_CH_BUF1_RDY_5	DMA_CH_BUF1_RDY_4	DMA_CH_BUF1_RDY_3	DMA_CH_BUF1_RDY_2	DMA_CH_BUF1_RDY_1	DMA_CH_BUF1_RDY_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-89. IPU Channels Buffer 1 Ready 0 Register (IPU_CH_BUF1_RDY0)

Table 42-86. IPU_CH_BUF1_RDY0 Field Descriptions

Field	Description
31–0 DMA_CH_BUF1_RDY_* [1]	Buffer 1 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.30 IPU Channels Buffer 1 Ready 1 Register (IPU_CH_BUF1_RDY1)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (63-32). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF1_RDY1_CLR bit.

The register is shown in Figure 42-90, and the register fields are described in Table 42-87.

Address 0xBASE+0xE000274 (IPU_CH_BUF1_RDY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_BUF1_RDY_52	DMA_CH_BUF1_RDY_51	DMA_CH_BUF1_RDY_50	DMA_CH_BUF1_RDY_49	DMA_CH_BUF1_RDY_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_CH_BUF1_RDY_47	DMA_CH_BUF1_RDY_46	DMA_CH_BUF1_RDY_45	DMA_CH_BUF1_RDY_44	DMA_CH_BUF1_RDY_43	DMA_CH_BUF1_RDY_42	DMA_CH_BUF1_RDY_41	DMA_CH_BUF1_RDY_40	0	0	0	0	0	0	DMA_CH_BUF1_RDY_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-90. IPU Channels Buffer 1 Ready 1 Register (IPU_CH_BUF1_RDY1)

Table 42-87. IPU_CH_BUF1_RDY1 Field Descriptions

Field	Description
31–0 DMA_CH_BUF1_RDY_*[1]	Buffer 1 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.31 IPU Alternate Channels Buffer 0 Ready 0 Register (IPU_ALT_CH_BUF0_RDY0)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (31-0). Writing "1" to each field will set each bit. Writing "0" to each field simultaneously will clear all the bits.

The register is shown in Figure 42-91, and the register fields are described in Table 42-88.

Address 0xBASE+0xE000278 (IPU_ALT_CH_BUF0_RDY0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	DMA_CH_ALT_BUF0_RDY_29	0	0	0	0	DMA_CH_ALT_BUF0_RDY_24	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DMA_CH_ALT_BUF0_RDY_7	DMA_CH_ALT_BUF0_RDY_6	DMA_CH_ALT_BUF0_RDY_5	DMA_CH_ALT_BUF0_RDY_4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-91. IPU Alternate Channels Buffer 0 Ready 0 Register (IPU_ALT_CH_BUF0_RDY0)

Table 42-88. IPU_ALT_CH_BUF0_RDY0 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_BUF0_RDY_*[1]	Buffer 0 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

1 Indicates the corresponding DMA channel number.

42.2.3.2.32 IPU Alternate Channels Buffer 0 Ready 1 Register (IPU_ALT_CH_BUF0_RDY1)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (63-32). Writing "1" to each field will set each bit. Writing "0" to each field simultaneously will clear all the bits.

The register is shown in Figure 42-80, and the register fields are described in Table 42-77.

Address 0xBASE+0xE00027C (IPU_ALT_CH_BUF0_RDY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_ALT_BUF0_RDY_52	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DMA_CH_ALT_BUF0_RDY_41	0	0	0	0	0	0	0	DMA_CH_ALT_BUF0_RDY_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-92. IPU Alternate Channels Buffer 0 Ready 1 Register (IPU_ALT_CH_BUF0_RDY1)

Table 42-89. IPU_ALT_CH_BUF0_RDY1 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_BUF0_RDY_*[1]	Buffer 0 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

1 Indicates the corresponding DMA channel number.

42.2.3.2.33 IPU Alternate Channels Buffer 1 Ready 0 Register (IPU_ALT_CH_BUF1_RDY0)

The register contains buffer 1 ready control information for 32 IPUv3's DMA channels (31-0). Writing "1" to each field will set each bit. Writing "0" to each field simultaneously will clear all the bits.

The register is shown in Figure 42-93, and the register fields are described in Table 42-90.

Address 0xBASE+0xE000280 (IPU_ALT_CH_BUF1_RDY0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	DMA_CH_ALT_BUF1_RDY_29	0	0	0	0	DMA_CH_ALT_BUF1_RDY_24	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DMA_CH_ALT_BUF1_RDY_7	DMA_CH_ALT_BUF1_RDY_6	DMA_CH_ALT_BUF1_RDY_5	DMA_CH_ALT_BUF1_RDY_4	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-93. IPU Alternate Channels Buffer 1 Ready 0 Register (IPU_ALT_CH_BUF1_RDY0)

Table 42-90. IPU_ALT_CH_BUF1_RDY0 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_BUF1_RDY_*[1]	Buffer 1 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

1 Indicates the corresponding DMA channel number.

42.2.3.2.34 IPU Alternate Channels Buffer 1 Ready 1 Register (IPU_ALT_CH_BUF1_RDY1)

The register contains buffer 0 ready control information for 32 IPUv3's DMA channels (63-32). Writing "1" to each field will set each bit. Writing "0" to each field simultaneously will clear all the bits.

The register is shown in Figure 42-94, and the register fields are described in Table 42-91.

Address 0xBASE+0xE000284 (IPU_ALT_CH_BUF1_RDY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	DMA_CH_ALT_BUF1_RDY_52	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DMA_CH_ALT_BUF1_RDY_41	0	0	0	0	0	0	0	DMA_CH_ALT_BUF1_RDY_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-94. IPU Alternate Channels Buffer 1 Ready 1 Register (IPU_ALT_CH_BUF1_RDY1)

Table 42-91. IPU_ALT_CH_BUF1_RDY1 Field Descriptions

Field	Description
31–0 DMA_CH_ALT_BUF1_RDY_*[1]	Buffer 1 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

1 Indicates the corresponding DMA channel number.

42.2.3.2.35 IPU Channels Buffer 2 Ready 0 Register (IPU_CH_BUF2_RDY0)

The register contains buffer 2 ready control information for 32 IPUv3's DMA channels (31-0). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF2_RDY0_CLR bit.

The register is shown in Figure 42-95, and the register fields are described in Table 42-92.

Address 0xBASE+0xE000288 (IPU_CH_BUF2_RDY0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	DMA_CH_BUF2_RDY_28	DMA_CH_BUF2_RDY_27	0	0	0	DMA_CH_BUF2_RDY_23	0	DMA_CH_BUF2_RDY_21	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	DMA_CH_BUF2_RDY_13	0	0	DMA_CH_BUF2_RDY_10	DMA_CH_BUF2_RDY_9	DMA_CH_BUF2_RDY_8	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-95. IPU Channels Buffer 2 Ready 0 Register (IPU_CH_BUF2_RDY0)

Table 42-92. IPU_CH_BUF2_RDY0 Field Descriptions

Field	Description
31–0 DMA_CH_BUF2_RDY_*[1]	Buffer 2 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 2 is not ready. 1 Buffer 2 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.2.36 IPU Channels Buffer 2 Ready 1 Register (IPU_CH_BUF2_RDY1)

The register contains buffer 2 ready control information for 32 IPUv3's DMA channels (63-32). This register can be a write one to set or a write one to clear according to the IPU_CH_BUF2_RDY1_CLR bit.

The register is shown in Figure 42-96, and the register fields are described in Table 42-93.

Address 0xBASE+0xE00028C (IPU_CH_BUF2_RDY1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-96. IPU Channels Buffer 2 Ready 1 Register (IPU_CH_BUF2_RDY1)

Table 42-93. IPU_CH_BUF2_RDY1 Field Descriptions

Field	Description
31–0 DMA_CH_BUF2_RDY_*[1]	Buffer 2 is ready. This bit indicates that MCU finished/writing reading buffer 0 in memory. 0 Buffer 0 is not ready. 1 Buffer 0 is ready.

[1] Indicates the corresponding DMA channel number.

42.2.3.3 IDMAC registers

42.2.3.3.1 IDMAC Configuration Register (IDMAC_CONF)

Address 0xBASE+0xE008000 (IDMAC_CONF) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	P_ENDIAN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	RDI	WIDPT		MAX_REQ_READ		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1

Figure 42-97. IDMAC Configuration Register (IDMAC_CONF)

Table 42-94. Register Field Descriptions

Field	Description
31–17	Reserved, should be cleared.
16 P_ENDIAN	Pixel Endianness. The pixel Endianness must not be changed while any of the IDMAC channels is enabled. 0 - little endian 1 - Big endian
15–6	Reserved, should be cleared.
5 RDI	Read Data Interleaving. This bit must match the slave read data interleaving support. If the AXI slave connected to the IPUv3EX supports read data interleaving then this bit must be set. If the AXI slave does not support read data interleaving then the IDMAC can utilize this and issue more address phases on read. In that case it is recommended to have this bit cleared. 0 - The AXI slave does not support read data interleaving 1 - The AXI slave supports read data interleaving
4–3 WIDPT	Write Interleaving Depth These 2 bits define the Write Interleaving Depth of the AXI port. This bits should be configured by the user according to the AXI slave's Write Interleaving Depth. WIDPT defines the maximal number of active bursts (yet to be responded) with different IDs. IDMAC will block data phase if the next data's ID is new (no such ID active) and the number of active IDs is equal to WIDPT. 00 Write Interleaving Depth of 1 01 Write Interleaving Depth of 2 10 Write Interleaving Depth of 3 11 Write Interleaving Depth of 4
2–0 MAX_REQ_READ	Maximum Read Requests. This fields sets the maximum pending requests allowed in the AXI Read requests queue.

42.2.3.3.2 IDMAC Channel Enable 1 Register (IDMAC_CH_EN_1)

Address 0xBASE+0xE008004 (IDMAC_CH_EN_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_CH_EN_31	0	IDMAC_CH_EN_29	IDMAC_CH_EN_28	IDMAC_CH_EN_27	0	0	IDMAC_CH_EN_24	IDMAC_CH_EN_23	IDMAC_CH_EN_22	IDMAC_CH_EN_21	IDMAC_CH_EN_20	0	IDMAC_CH_EN_18	IDMAC_CH_EN_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_EN_15	IDMAC_CH_EN_14	IDMAC_CH_EN_13	IDMAC_CH_EN_12	IDMAC_CH_EN_11	IDMAC_CH_EN_10	IDMAC_CH_EN_9	IDMAC_CH_EN_8	IDMAC_CH_EN_7	IDMAC_CH_EN_6	IDMAC_CH_EN_5	IDMAC_CH_EN_4	IDMAC_CH_EN_3	IDMAC_CH_EN_2	IDMAC_CH_EN_1	IDMAC_CH_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 42-95. Register Field Descriptions

Field	Description
IDMAC_CH_EN_<i>	IDMAC Channel enable bit [i] 0 - IDMAC channel is disabled 1 - IDMAC channel is enabled

42.2.3.3.3 IDMAC Channel Enable 2 Register (IDMAC_CH_EN_2)

Address 0xBASE+0xE008008 (IDMAC_CH_EN_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_CH_EN_52	IDMAC_CH_EN_51	IDMAC_CH_EN_50	IDMAC_CH_EN_49	IDMAC_CH_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_EN_47	IDMAC_CH_EN_46	IDMAC_CH_EN_45	IDMAC_CH_EN_44	IDMAC_CH_EN_43	IDMAC_CH_EN_42	IDMAC_CH_EN_41	IDMAC_CH_EN_40	0	0	0	0	0	0	IDMAC_CH_EN_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-98. IDMAC Channel Enable 2 Register (IDMAC_CH_EN_2)

Table 42-96. Register Field Descriptions

Field	Description
IDMAC_CH_EN_<i>	IDMAC Channel enable bit [i] 0 - IDMAC channel is disabled 1 - IDMAC channel is enabled

42.2.3.3.4 IDMAC Separate Alpha Indication Register (IDMAC_SEP_ALPHA)

Address 0xBASE+0xE00800C (IDMAC_SEP_ALPHA) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	IDMAC_SEP_AL_29	0	IDMAC_SEP_AL_27	0	0	IDMAC_SEP_AL_24	IDMAC_SEP_AL_23	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_SEP_AL_15	IDMAC_SEP_AL_14	0	0	0	IDMAC_SEP_AL_10	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-99. IDMAC Separate Alpha Indication Register (IDMAC_SEP_ALPHA)

Table 42-97. Register Field Descriptions

Field	Description
IDMAC_SEP_AL_<i>	IDMAC Separate alpha indication bit [i] A sub module may need to read data from the system's memory where the pixel data and the alpha transparency data are located in separate buffers. In that case the Alpha transparency data is read by the special Alpha channel. In a case where the alpha should be read from a separate buffer, the user should set the channels corresponding. 0 - Channel [i] does not read Alpha transparency data from a separate buffer. 1 - Channel [i] reads Alpha transparency data from a separate buffer.

42.2.3.3.5 IDMAC Alternate Separate Alpha Indication Register (IDMAC_ALT_SEP_ALPHA)

Address 0xBASE+0xE008010 (IDMAC_ALT_SEP_ALPHA) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	IDMAC_ALT_SEP_AL_29	0	0	0	0	IDMAC_ALT_SEP_AL_24	IDMAC_ALT_SEP_AL_23	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-100. IDMAC Alternate Separate Alpha Indication Register (IDMAC_ALT_SEP_ALPHA)

Table 42-98. Register Field Descriptions

Field	Description
IDMAC_ALT_SEP_AL_<i>	IDMAC Alternate Separate alpha indication bit [i] A sub module may need to read data from the system's memory where the pixel data and the alpha transparency data are located in separate buffers. In that case the Alpha transparency data is read by the special Alpha channel. In a case where the alpha should be read from a separate buffer, the user should set the channels corresponding. For channels that may have alternate flow and may use separate alpha. This bit indicates the mode of the alpha for the alternate flow 0 - Channel [i] does not read Alpha transparency data from a separate buffer. 1 - Channel [i] reads Alpha transparency data from a separate buffer.

42.2.3.3.6 IDMAC Channel Priority 1 Register (IDMAC_CH_PRI_1)

Address 0xBASE+0xE008014 (IDMAC_CH_PRI_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	IDMAC_CH_PRI_29	IDMAC_CH_PRI_28	IDMAC_CH_PRI_27	0	0	IDMAC_CH_PRI_24	IDMAC_CH_PRI_23	IDMAC_CH_PRI_22	IDMAC_CH_PRI_21	IDMAC_CH_PRI_20	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_PRI_15	IDMAC_CH_PRI_14	IDMAC_CH_PRI_13	IDMAC_CH_PRI_12	IDMAC_CH_PRI_11	IDMAC_CH_PRI_10	IDMAC_CH_PRI_9	IDMAC_CH_PRI_8	IDMAC_CH_PRI_7	IDMAC_CH_PRI_6	IDMAC_CH_PRI_5	IDMAC_CH_PRI_4	IDMAC_CH_PRI_3	IDMAC_CH_PRI_2	IDMAC_CH_PRI_1	IDMAC_CH_PRI_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 42-99. Register Field Descriptions

Field	Description
IDMAC_CH_PRI_<i>	IDMAC Channel enable bit [i] 0 - IDMAC channel [i] is in low priority 1 - IDMAC channel [i] is in high priority

42.2.3.3.7 IDMAC Channel Priority 2 Register (IDMAC_CH_PRI_2)

Address 0xBASE+0xE008018 (IDMAC_CH_PRI_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_CH_PRI_50	IDMAC_CH_PRI_49	IDMAC_CH_PRI_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_PRI_47	IDMAC_CH_PRI_46	IDMAC_CH_PRI_45	IDMAC_CH_PRI_44	IDMAC_CH_PRI_43	IDMAC_CH_PRI_42	IDMAC_CH_PRI_41	IDMAC_CH_PRI_40	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-101. IDMAC Channel Priority 2 Register (IDMAC_CH_PRI_2)

Table 42-100. Register Field Descriptions

Field	Description
IDMAC_CH_PRI_<i>	IDMAC Channel enable bit [i] 0 - IDMAC channel [i] is in low priority 1 - IDMAC channel [i] is in high priority

42.2.3.3.8 IDMAC Channel Watermark Enable 1 Register (IDMAC_WM_EN_1)

Address 0xBASE+0xE00801C (IDMAC_WM_EN_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	IDMAC_WM_EN_29	IDMAC_WM_EN_28	IDMAC_WM_EN_27	0	0	IDMAC_WM_EN_24	IDMAC_WM_EN_23	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_WM_EN_14	IDMAC_WM_EN_13	IDMAC_WM_EN_12	0	IDMAC_WM_EN_10	0	IDMAC_WM_EN_8	0	0	0	0	IDMAC_WM_EN_3	IDMAC_WM_EN_2	IDMAC_WM_EN_1	IDMAC_WM_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 42-101. Register Field Descriptions

Field	Description
IDMAC_WM_EN_<i>	IDMAC Watermark enable bit [i] 0 - IDMAC channel [i] watermark feature is disabled 1 - IDMAC channel [i] watermark feature is enabled

42.2.3.3.9 IDMAC Channel Watermark Enable 2 Register (IDMAC_WM_EN_2)

Address 0xBASE+0xE008020 (IDMAC_WM_EN_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	IDMAC_WM_EN_44	IDMAC_WM_EN_43	IDMAC_WM_EN_42	IDMAC_WM_EN_41	IDMAC_WM_EN_40	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-102. IDMAC Channel Watermark Enable 2 Register (IDMAC_WM_EN_2)

Table 42-102. Register Field Descriptions

Field	Description
IDMAC_WM_EN_<i>	IDMAC Watermark enable bit [i] 0 - IDMAC channel [i] watermark feature is disabled 1 - IDMAC channel [i] watermark feature is enabled

42.2.3.3.10 IDMAC Channel Lock Enable 1 Register (IDMAC_LOCK_EN_1)

Address 0xBASE+0xE008024 (IDMAC_LOCK_EN_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	IDMAC_LOCK_EN_22	IDMAC_LOCK_EN_21	IDMAC_LOCK_EN_20	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_LOCK_EN_15	IDMAC_LOCK_EN_14	0	IDMAC_LOCK_EN_12	IDMAC_LOCK_EN_11	0	0	0	0	0	IDMAC_LOCK_EN_5	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-103. IDMAC Channel Lock Enable 1 Register (IDMAC_LOCK_EN_1)

42.2.3.3.11 IDMAC Channel Lock Enable 2 Register (IDMAC_LOCK_EN_2)

Address 0xBASE+0xE008028 (IDMAC_LOCK_EN_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_LOCK_EN_50	IDMAC_LOCK_EN_49	IDMAC_LOCK_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_LOCK_EN_47	IDMAC_LOCK_EN_46	IDMAC_LOCK_EN_45	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-104. IDMAC Channel Lock Enable 2 Register (IDMAC_LOCK_EN_2)

Table 42-104. Register Field Descriptions

Field	Description
IDMAC_LOCK_EN_<i>	IDMAC lock enable bit [i] 0 - IDMAC channel [i] lock feature is disabled 1 - IDMAC channel [i] lock feature is enabled

42.2.3.3.12 IDMAC Channel Alternate address 0 Register (IDMAC_SUB_ADDR_0)

Address 0xBASE+0xE00802C (IDMAC_SUB_ADDR_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	IDMAC_SUB_ADDR_7							0	IDMAC_SUB_ADDR_6						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_SUB_ADDR_5							0	IDMAC_SUB_ADDR_4						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-105. IDMAC Channel Alternate address 0 Register (IDMAC_SUB_ADDR_0)

Table 42-105. Register Field Descriptions

Field	Description
IDMAC_SUB_ADDR[i]	The CPMEM alternative entry [i] holds the parameters of the channel that is number appears in IDMAC_SUB_ADDR[i]. The user must set each IDMAC_SUB_ADDR[i] field to a unique channel. Alternative CPMEM entry is relevant only for channels supporting alternate flows.

42.2.3.3.13 IDMAC Channel Alternate address 1 Register (IDMAC_SUB_ADDR_1)

Address 0xBASE+0xE008030 (IDMAC_SUB_ADDR_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	IDMAC_SUB_ADDR_33							0	IDMAC_SUB_ADDR_29						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_SUB_ADDR_24							0	IDMAC_SUB_ADDR_23						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-106. IDMAC Channel Alternate address 1 Register (IDMAC_SUB_ADDR_1)

Table 42-106. Register Field Descriptions

Field	Description
IDMAC_SUB_ADDR[i]	The CPMEM alternative entry [i] holds the parameters of the channel that is number appears in IDMAC_SUB_ADDR[i]. The user must set each IDMAC_SUB_ADDR[i] field to a unique channel. Alternative CPMEM entry is relevant only for channels supporting alternate flows.

42.2.3.3.14 IDMAC Channel Alternate address 2 Register (IDMAC_SUB_ADDR_2)

Address 0xBASE+0xE008034 (IDMAC_SUB_ADDR_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	IDMAC_SUB_ADDR_52						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_SUB_ADDR_51							0	IDMAC_SUB_ADDR_41						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-107. IDMAC Channel Alternate address 2 Register (IDMAC_SUB_ADDR_2)

Table 42-107. Register Field Descriptions

Field	Description
IDMAC_SUB_ADDR[i]	The CPMEM alternative entry [i] holds the parameters of the channel that is number appears in IDMAC_SUB_ADDR[i]. The user must set each IDMAC_SUB_ADDR[i] field to a unique channel. Alternative CPMEM entry is relevant only for channels supporting alternate flows.

42.2.3.3.15 IDMAC Channel Alternate address 3 Register (IDMAC_SUB_ADDR_3)

Address 0xBASE+0xE008038 (IDMAC_SUB_ADDR_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	IDMAC_SUB_ADDR_27							0	IDMAC_SUB_ADDR_13						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_SUB_ADDR_10							0	IDMAC_SUB_ADDR_9						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-108. IDMAC Channel Alternate address 3 Register (IDMAC_SUB_ADDR_3)

Table 42-108. Register Field Descriptions

Field	Description
IDMAC_SUB_ADDR[i]	The CPMEM alternative entry [i] holds the parameters of the channel that is number appears in IDMAC_SUB_ADDR[i]. The user must set each IDMAC_SUB_ADDR[i] field to a unique channel. Alternative CPMEM entry is relevant only for channels supporting alternate flows.

42.2.3.3.16 IDMAC Channel Alternate address 4 Register (IDMAC_SUB_ADDR_4)

Address 0xBASE+0xE00803C (IDMAC_SUB_ADDR_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	IDMAC_SUB_ADDR_21						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IDMAC_SUB_ADDR_8							0	IDMAC_SUB_ADDR_28						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-109. IDMAC Channel Alternate address 4 Register (IDMAC_SUB_ADDR_4)

Table 42-109. Register Field Descriptions

Field	Description
IDMAC_SUB_ADDR[i]	The CPMEM alternative entry [i] holds the parameters of the channel that is number appears in IDMAC_SUB_ADDR[i]. The user must set each IDMAC_SUB_ADDR[i] field to a unique channel. Alternative CPMEM entry is relevant only for channels supporting alternate flows.

42.2.3.3.17 IDMAC Band Mode Enable 1 Register (IDMAC_BNDM_EN_1)

Address 0xBASE+0xE008040 (IDMAC_BNDM_EN_1) Access: User read-write

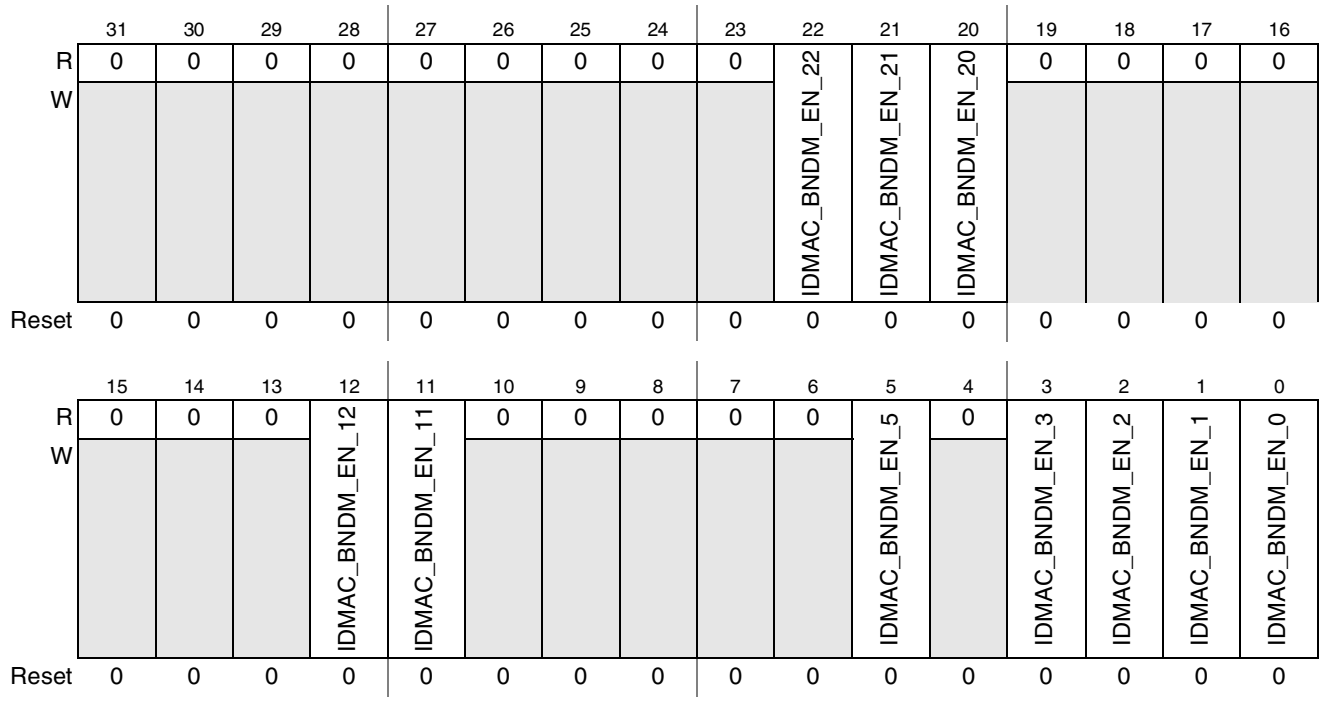


Figure 42-110. IDMAC Band Mode Enable 1 Register (IDMAC_BNDM_EN_1)

Table 42-110. Register Field Descriptions

Field	Description
IDMAC_ BNDM_ EN_<i>	IDMAC Band Mode Enable bit [i] This bit controls if the channel currently works in band mode. When alternate flow is running via this channel, both the main flow and the alternate flow should have the same band mode configuration. 0 - IDMAC channel [i] is not in band mode 1 - IDMAC channel [i] is in band mode

42.2.3.3.18 IDMAC Band Mode Enable 2 Register (IDMAC_BNDM_EN_2)

Address 0xBASE+0xE008044 (IDMAC_BNDM_EN_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	IDMAC_BNDM_EN_50	IDMAC_BNDM_EN_49	IDMAC_BNDM_EN_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_BNDM_EN_47	IDMAC_BNDM_EN_46	IDMAC_BNDM_EN_45	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-111. IDMAC Band Mode Enable 2 Register (IDMAC_BNDM_EN_2)

Table 42-111. Register Field Descriptions

Field	Description
IDMAC_ BNDM_ EN_<i>	IDMAC Band Mode Enable bit [i] This bit controls if the channel currently works in band mode. When alternate flow is running via this channel, both the main flow and the alternate flow should have the same band mode configuration. 0 - IDMAC channel [i] is not in band mode 1 - IDMAC channel [i] is in band mode

42.2.3.3.19 IDMAC Scroll Coordinations Register (IDMAC_SC_CORD)

Address 0xBASE+0xE008048 (IDMAC_SC_CORD) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	SX0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	SY0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-112. IDMAC Scroll Coordinations Register (IDMAC_SC_CORD)

Table 42-112. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 SX0	Scroll X coordination This field indicates the X coordinate of the scroll. This parameter has an affect on continuos scroll mode only. Units are pixels. When the alpha buffer resides in a separate buffer and the alpha is different than 8 BPP this field should be 0
15–11	Reserved, should be cleared.
10–0 SY0	Scroll Y coordination This field indicates the Y coordinate of the scroll. This parameter has an affect on continuos scroll mode only. Units are pixels. When the alpha buffer resides in a separate buffer and the alpha is different than 8 BPP this field should be 0

42.2.3.3.20 IDMAC Scroll Coordinations 1 Register (IDMAC_SC_CORD1)

Address 0xBASE+0xE00804C (IDMAC_SC_CORD1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	SX1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	SY1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-113. IDMAC Scroll Coordinations 1 Register (IDMAC_SC_CORD1)

Table 42-113. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 SX1	Scroll X coordination (2nd set) This field indicates the X coordinate of the scroll. This parameter has an affect on continuos scroll mode only. Units are pixels. When the alpha buffer resides in a separate buffer and the alpha is different than 8 BPP this field should be 0
15–11	Reserved, should be cleared.
10–0 SY1	Scroll Y coordination (2nd set) This field indicates the Y coordinate of the scroll. This parameter has an affect on continuos scroll mode only. Units are pixels. When the alpha buffer resides in a separate buffer and the alpha is different than 8 BPP this field should be 0

42.2.3.3.21 IDMAC Channel Busy 1 Register (IDMAC_CH_BUSY_1)

Address 0xBASE+0xE008100 (IDMAC_CH_BUSY_1) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IDMAC_CH_BUSY_31	0	IDMAC_CH_BUSY_29	IDMAC_CH_BUSY_28	IDMAC_CH_BUSY_27	0	0	IDMAC_CH_BUSY_24	IDMAC_CH_BUSY_23	IDMAC_CH_BUSY_22	IDMAC_CH_BUSY_21	IDMAC_CH_BUSY_20	0	IDMAC_CH_BUSY_18	IDMAC_CH_BUSY_17	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_BUSY_15	IDMAC_CH_BUSY_14	IDMAC_CH_BUSY_13	IDMAC_CH_BUSY_12	IDMAC_CH_BUSY_11	IDMAC_CH_BUSY_10	IDMAC_CH_BUSY_9	IDMAC_CH_BUSY_8	IDMAC_CH_BUSY_7	IDMAC_CH_BUSY_6	IDMAC_CH_BUSY_5	IDMAC_CH_BUSY_4	IDMAC_CH_BUSY_3	IDMAC_CH_BUSY_2	IDMAC_CH_BUSY_1	IDMAC_CH_BUSY_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 42-114. Register Field Descriptions

Field	Description
IDMAC_CH_BUSY_<i>	IDMAC Channel busy bit [i] This bit indicates if the channel is currently served by the IDMAC. 0 - IDMAC channel [i] is not busy 1 - IDMAC channel [i] is busy This bit is self cleared by the IDMAC

42.2.3.3.22 IDMAC Channel Busy 2 Register (IDMAC_CH_BUSY_2)

Address 0xBASE+0xE008104 (IDMAC_CH_BUSY_2) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	IDMAC_CH_BUSY_52	IDMAC_CH_BUSY_51	IDMAC_CH_BUSY_50	IDMAC_CH_BUSY_49	IDMAC_CH_BUSY_48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IDMAC_CH_BUSY_47	IDMAC_CH_BUSY_46	IDMAC_CH_BUSY_45	IDMAC_CH_BUSY_44	IDMAC_CH_BUSY_43	IDMAC_CH_BUSY_42	IDMAC_CH_BUSY_41	IDMAC_CH_BUSY_40	0	0	0	0	0	0	IDMAC_CH_BUSY_33	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-114. IDMAC Channel Busy 2 Register (IDMAC_CH_BUSY_2)

Table 42-115. Register Field Descriptions

Field	Description
IDMAC_CH_BUSY_<i>	IDMAC Channel busy bit [i] This bit indicates if the channel is currently served by the IDMAC. 0 - IDMAC channel [i] is not busy 1 - IDMAC channel [i] is busy This bit is self cleared by the IDMAC

42.2.3.4 DP Registers

42.2.3.4.1 DP Common Configuration Sync Flow Register (DP_COM_CONF_SYNC)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF040000 (DP_COM_CONF_SYNC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	DP_GAMMA_YUV_EN_SYNC	DP_GAMMA_EN_SYNC	DP_CSC_YUV_SAT_MODE_SYNC	DP_CSC_GAMUT_SAT_EN_SYNC	DP_CSC_DEF_SYNC		0	DP_COC_SYNC			DP_GWCKE_SYNC	DP_GWAM_SYNC	DP_GWSEL_SYNC	DP_FG_EN_SYNC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-116. DP Common Configuration Sync Flow Register (DP_COM_CONF_SYNC)

Table 42-116. DP_COM_CONF_SYNC Field Descriptions

Field	Description
31–14	Reserved
13 DP_GAMMA_YUV_EN_SYNC	GAMMA's YUV mode enable for sync flow 0 YUV mode is OFF. 1 YUV mode is ON.
12 DP_GAMMA_EN_SYNC	GAMMA_EN - Gamma correction module enable bit 0 disable 1 enable
11 DP_CSC_YUV_SAT_MODE_SYNC	CSC_YUV_SAT_MODE YUV saturation mode for color space conversion 0 - Y/U/V range 0 -255 1- Y range 16-235, U/V range 16-240
10 DP_CSC_GAMUT_SAT_EN_SYNC	CSC_GAMUT_SAT_EN Indicate if GAMUT saturation is enabled 0 - disable GAMUT mapping 1 - enable GAMUT mapping

Table 42-116. DP_COM_CONF_SYNC Field Descriptions (continued)

Field	Description
9–8 DP_CSC_DEF_SYNC	CSC_DEF Enable or disable Color Space Conversion. 00 CSC disable 01 CSC enable after combining 10 CSC enable before combining on BG channel 11 CSC enable before combining on FG channel
7	Reserved
6–4 DP_COC_SYNC	COC - Cursor Operation Control Controls the format of the cursor and the type of arithmetic operations 000Transparent, cursor is disabled. 001Full cursor. 010Reversed cursor. 011AND between full plane and cursor. 100 Reserved 101OR between full plane and cursor. 110XOR between full plane and cursor. 111Reserved.
3 DP_GWCKE_SYNC	GWCKE - Graphic Window Color Keying Enable Enable or disable graphic window color keying. 1 Enable color keying of graphic window 0 Disable color keying of graphic window
2 DP_GWAM_SYNC	GWAM - Graphic Window Alpha Mode Select the use of Alpha to be global or local. 1 Global Alpha. 0 Local Alpha.
1 DP_GWSEL_SYNC	GWSEL - Graphic Window Select Select graphic window to be on partial plane or full plane. 1 Graphic window is partial plane. 0 Graphic window is full plane.
0 DP_FG_EN_SYNC	FG_EN - partial plane Enable. This bit enables the partial plane channel. 1 partial plane channel is enabled. 0 partial plane channel is disabled.

42.2.3.4.2 DP Graphic Window Control Sync Flow Register (DP_GRAPH_WIND_CTRL_SYNC)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF040004 (DP_GRAPH_WIND_CTRL_SYNC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_GWAV_SYNC								DP_GWCKR_SYNC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_GWCKG_SYNC								DP_GWCKB_SYNC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-117. DP Graphic Window Control Sync Flow Register (DP_GRAPH_WIND_CTRL_SYNC)

Table 42-117. DP_GRAPH_WIND_CTRL_SYNC Field Descriptions

Field	Description
31–23 DP_GWAV_SYNC	GWAV - Graphic Window Alpha Value Defines the alpha value of graphic window used for alpha blending between graphic window and full plane. The Value the number that writing to GWAV register. The Actual Value the number, that insert to calculation in Combining Module. If Value = < 0.5- 1/256 (01111111) then Actual Value = Value. If Value >= 0.5 (10000000), then Actual Value = Value + 1/256. 00000000 Actual value is 00000000; Graphic window totally opaque i.e. overlay on LCD screen ..... 01111111Actual value is 01111111; 10000000Actual value is 10000001 10000001Actual value is 10000010 ..... 11111110Actual value is 11111111 11111111Actual value is 100000000;Graphic window totally transparent i.e. not displayed on LCD screen
31–23 DP_GWCKR_SYNC	GWCKR - Graphic Window Color Keying Red Component Defines the red component of graphic window color keying. 00000000No red ..... 11111111Full red

Table 42-117. DP_GRAPH_WIND_CTRL_SYNC Field Descriptions (continued)

Field	Description
31–23 DP_GWCKG_SYNC	GWCKG - Graphic Window Color Keying Green Component Defines the green component of graphic window color keying. 00000000No Green ..... 11111111Full Green
31–23 DP_GWCKB_SYNC	GWCKB - Graphic Window Color Keying Blue Component Defines the blue component of graphic window color keying. 00000000No blue ..... 11111111Full blue

42.2.3.4.3 DP partial plane Window Position Sync Flow Register (DP_FG_POS_SYNC)

The DP partial plane Window Position Register is used to determine the starting position of the partial plane window relative to BG window position.

This Register is used for the synchronous flow only.

Address 0xBASE+0xF040008 (DP_FG_POS_SYNC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	DP_FGXP_SYNC										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	DP_FGYP_SYNC										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-118. DP partial plane Window Position Sync Flow Register (DP_FG_POS_SYNC)

Table 42-118. DP_FG_POS_SYNC Field Descriptions

Field	Description
31–27	Reserved
26–16 DP_FGXP_SYNC	FGXP partial plane Window X Position. partial plane Window X Position - Specifies the number of pixels between the start of full plane window X position and the beginning of the first data of new line.

Table 42-118. DP_FG_POS_SYNC Field Descriptions (continued)

Field	Description
15–11	Reserved
10–0 DP_FGYP_SYNC	FGYP partial plane Window Y Position partial plane Window Y Position - Specifies the number of lines between the start of full plane windows Y position and the beginning of the first data.

42.2.3.4.4 DP Cursor Position and Size Sync Flow Register (DP_CUR_POS_SYNC)

The LCD Cursor Position Register is used to determine the starting position of the cursor relative to BG windows position.

Address 0xBASE+0xF04000C (DP_CUR_POS_SYNC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CXW_SYNC								DP_CXP_SYNC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CYH_SYNC								DP_CYP_SYNC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-119. DP Cursor Position and Size Sync Flow Register (DP_CUR_POS_SYNC)

Table 42-119. DP_CUR_POS_SYNC Field Descriptions

Field	Description
31–27 DP_CYP_SYNC	CYP - Cursor Y Position Represents the cursors vertical starting position Y in pixel count (from 0 to CYH).Live View Resolution Mode.
26–16 DP_CYH_SYNC	CYH - Cursor Height Specifies the height of the hardware cursor in pixels.
15–11 DP_CXP_SYNC	CXP - Cursor X Position Represents the cursors horizontal starting position X in pixel count (from 0 to CXW).
10–0 DP_CXW_SYNC	CXW - Cursor Width. Specifies the width of the hardware cursor in pixels.

42.2.3.4.5 DP Color Cursor Mapping Sync Flow Register (DP_CUR_MAP_SYNC)

The LCD Color Cursor Mapping Register defines the color of the cursor in passive or TFT color modes.

Address 0xBASE+0xF040010 (DP_CUR_MAP_SYNC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	DP_CUR_COL_R_SYNC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DP_CUR_COL_G_SYNC								DP_CUR_COL_B_SYNC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-120. DP Cursor Position and Size Sync Flow Register (DP_CUR_POS_SYNC)

Table 42-120. DP_CUR_POS_SYNC Field Descriptions

Field	Description
23–16 DP_CUR_COL_B_SYNC	CUR_COL_B - Cursor Blue Field Defines the Blue component of the cursor color in color mode 00000000No Blue. .... 11111111Full Blue.
15–8 DP_CUR_COL_G_SYNC	CUR_COL_G - Cursor Green Field Defines the Green component of the cursor color in color mode 00000000No Green. ..... 11111111Full Green.
7–0 DP_CUR_COL_B_SYNC	CUR_COL_B - Cursor Red Field Defines the Red component of the cursor color in color mode 00000000No Red. ..... 11111111Full Red.

42.2.3.4.6 DP Gamma Constants Sync Flow Register i (DP_GAMMA_C_SYNC_i)

This registers contains CONSTANTi parameters used for gamma correction inside the display processor (DP).

Address 0xBASE+0xF040014 (DP_GAMMA_C_SYNC_<i>)
offset 4
range 0 .. 7

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								DP_GAMMA_C_SYNC_<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								DP_GAMMA_C_SYNC_<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-121. DP Gamma Constants Sync Flow Register i (DP_GAMMA_C_SYNC_i)

Table 42-121. DP_GAMMA_C_SYNC_i Field Descriptions

Field	Description
31-28	Reserved.
27-16	CONSTANTi+1 parameter of Gamma Correction.
15-9	Reserved.
8-0	CONSTANTi parameter of Gamma Correction.

42.2.3.4.7 DP Gamma Correction Slope Sync Flow Register i (DP_GAMMA_S_SYNC_i)

This registers contains SLOPEi parameters used for Gamma Correction inside the display processor (DP).

Address 0xBASE+0xF040034 (DP_GAMMA_S_SYNC_<i>)
offset 4
range 0 .. 3

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DP_GAMMA_S_SYNC_<4*i+3>								DP_GAMMA_S_SYNC_<4*i+2>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DP_GAMMA_S_SYNC_<4*i+1>								DP_GAMMA_S_SYNC_<4*i>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-122. DP Gamma Correction Slope Sync Flow Register i (DP_GAMMA_S_SYNC_i)

Table 42-122. DP_GAMMA_S_SYNC_i Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Gamma Correction.
23-16	SLOPE<4*i+2> parameter of Gamma Correction.
15-8	SLOPE<4*i+1> parameter of Gamma Correction.
7-0	SLOPE<4*i> parameter of Gamma Correction.

42.2.3.4.8 DP Color Space Conversion Control Sync Flow registers (DP_CSCA_SYNC_i).

Address 0xBASE+0xF040044 (DP_CSCA_SYNC_<i>)
offset 4
range 0 .. 3

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R							DP_CSC_A_SYNC_<2*i+1>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A_SYNC_<2*i>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-123. DP Color Space Conversion Control Sync Flow registers (DP_CSCA_SYNC_i).

Table 42-123. DP_CSCA_SYNC_i Field Descriptions

Field	Description
31-26	Reserved.
25-16 DP_CSC_A_SYNC<2*i+1>	A<2*i+1> parameter of color conversion
15-10	Reserved.
9-0 DP_CSC_A_SYNC<2*i>	A<2*i> parameter of color conversion.

42.2.3.4.9 DP Color Conversion Control Sync Flow registers (DP_CSC_SYNC_0).

Address 0xBASE+0xF040054 (DP_CSC_SYNC_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DP_CSC_S0_SYNC				DP_CSC_B0_SYNC											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A8_SYNC									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-124. DP Color Conversion Control Sync Flow registers (DP_CSC_SYNC_0).

Table 42-124. DP_CSC_SYNC_0 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-10	Reserved.
9-0	A9 parameter of color conversion.

42.2.3.4.10 DP Color Conversion Control Sync Flow registers (DP_CSC_SYNC_1).

Address 0xBASE+0xF040058 (DP_CSC_SYNC_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DP_CSC_S2_SYNC				DP_CSC_B2_SYNC											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DP_CSC_S1_SYNC				DP_CSC_B1_SYNC											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-125. DP Color Conversion Control Sync Flow registers (DP_CSC_SYNC_1).

Table 42-125. DP_CSC_SYNC_1 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-14	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
13-0	B0 parameter of color conversion.

42.2.3.4.11 DP Cursor Position and Size Alternate Register (DP_CUR_POS_ALT)

The LCD Cursor Position Register is used to determine the starting position of the cursor relative to BG windows position for the alternative flow.

Address 0xBASE+0xF04005C (DP_CUR_POS_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CXW_SYNC_ALT					DP_CXP_SYNC_ALT										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CYH_SYNC_ALT					DP_CYP_SYNC_ALT										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-126. DP Cursor Position and Size Alternate Register (DP_CUR_POS_ALT)

Table 42-126. DP_CUR_POS_ALT Field Descriptions

Field	Description
31–27 DP_CYP_SYNC_ALT	CYP_ALT - Cursor Y Position Represents the cursors vertical starting position Y in pixel count (from 0 to CYH).Live View Resolution Mode for the alternative flow.
26–16 DP_CYH_SYNC_ALT	CYH_ALT - Cursor Height Specifies the height of the hardware cursor in pixels.

Table 42-126. DP_CUR_POS_ALT Field Descriptions (continued)

Field	Description
15–11 DP_CXP_SYNC_ALT	CXP_ALT - Cursor X Position Represents the cursors horizontal starting position X in pixel count (from 0 to CXW) for the alternative flow.
10–0 DP_CXW_SYNC_ALT	CXW_ALT - Cursor Width. Specifies the width of the hardware cursor in pixels for the alternative flow.

42.2.3.4.12 DP Common Configuration async0 Flow Register (DP_COM_CONF_ASYNC0)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF040060 (DP_COM_CONF_ASYNC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	DP_GAMMA_YUV_EN_ASYNC0	DP_GAMMA_EN_ASYNC0	DP_CSC_YUV_SAT_MODE_ASYNC0	DP_CSC_GAMUT_SAT_EN_ASYNC0	DP_CSC_DEF_ASYNC0		0	DP_COC_ASYNC0			DP_GWCKE_ASYNC0	DP_GWAM_ASYNC0	DP_GWSEL_ASYNC0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-127. DP Common Configuration async0 Flow Register (DP_COM_CONF_ASYNC0)

Table 42-127. DP_COM_CONF_ASYNC0 Field Descriptions

Field	Description
31–14	Reserved
13 DP_GAMMA_YUV_EN_ASYNC0	GAMMA's YUV mode enable for async flow 0 0 YUV mode is OFF. 1 YUV mode is ON.
12 DP_GAMMA_EN_ASYNC0	GAMMA_EN - Gamma correction module enable bit 0 disable 1 enable
11 DP_CSC_YUV_SAT_MODE_ASYNC0	CSC_YUV_SAT_MODE YUV saturation mode for color space conversion 0 - Y/U/V range 0 -255 1- Y range 16-235, U/V range 16-240
10 DP_CSC_GAMUT_SAT_EN_ASYNC0	CSC_GAMUT_SAT_EN Indicate if GAMUT saturation is enabled 0 - disable GAMUT mapping 1 - enable GAMUT mapping
9–8 DP_CSC_DEF_ASYNC0	CSC_DEF Enable or disable Color Space Conversion. 00 CSC disable 01 CSC enable after combining 10 CSC enable before combining on BG channel 11 CSC enable before combining on FG channel
6–4 DP_COC_ASYNC0	COC - Cursor Operation Control Controls the format of the cursor and the type of arithmetic operations 000Transparent, cursor is disabled. 001Full cursor. 010Reversed cursor. 011AND between full plane and cursor. 100 Reserved 101OR between full plane and cursor. 110XOR between full plane and cursor. 111Reserved
3 DP_GWCKE_ASYNC0	GWCKE - Graphic Window Color Keying Enable Enable or disable graphic window color keying. 1 Enable color keying of graphic window 0 Disable color keying of graphic window
2 DP_GWAM_ASYNC0	GWAM - Graphic Window Alpha Mode Select the use of Alpha to be global or local. 1 Global Alpha. 0 Local Alpha.
1 DP_GWSEL_ASYNC0	GWSEL - Graphic Window Select Select graphic window to be on partial plane or full plane. 1 Graphic window is partial plane. 0 Graphic window is full plane.5
0	Reserved

42.2.3.4.13 DP Graphic Window Control async0 Flow Register (DP_GRAPH_WIND_CTRL_ASYNC0)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF040064 (DP_GRAPH_WIND_CTRL_ASYNC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_GWAV_ASYNC0								DP_GWCKR_ASYNC0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_GWCKG_ASYNC0								DP_GWCKB_ASYNC0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-128. DP Graphic Window Control async0 Flow Register (DP_GRAPH_WIND_CTRL_ASYNC0)

Table 42-128. DP_GRAPH_WIND_CTRL_ASYNC0 Field Descriptions

Field	Description
31–23 DP_GWAV_ASYNC0	GWAV - Graphic Window Alpha Value Defines the alpha value of graphic window used for alpha blending between graphic window and full plane. The Value the number that writing to GWAV register. The Actual Value the number, that insert to calculation in Combining Module. If Value = < 0.5- 1/256 (01111111) then Actual Value = Value. If Value >= 0.5 (10000000), then Actual Value = Value + 1/256. 00000000 Actual value is 00000000; Graphic window totally opaque i.e. overlay on LCD screen ..... 01111111Actual value is 01111111; 10000000Actual value is 10000001 10000001Actual value is 10000010 ..... 11111110Actual value is 11111111 11111111Actual value is 100000000;Graphic window totally transparent i.e. not displayed on LCD screen
31–23 DP_GWCKR_ASYNC0	GWCKR - Graphic Window Color Keying Red Component Defines the red component of graphic window color keying. 00000000No red ..... 11111111Full red

Table 42-128. DP_GRAPH_WIND_CTRL_ASYNC0 Field Descriptions (continued)

Field	Description
31–23 DP_GWCKG_ASYNC0	GWCKG - Graphic Window Color Keying Green Component Defines the green component of graphic window color keying. 00000000No Green ..... 11111111Full Green
31–23 DP_GWCKB_ASYNC0	GWCKB - Graphic Window Color Keying Blue Component Defines the blue component of graphic window color keying. 00000000No blue ..... 11111111Full blue

42.2.3.4.14 DP partial plane Window Position async0 Flow Register (DP_FG_POS_ASYNC0)

The DP partial plane Window Position Register is used to determine the starting position of the partial plane window relative to BG window position.

This Register is used for the synchronous flow only.

Address 0xBASE+0xF040068 (DP_FG_POS_ASYNC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	DP_FGXP_ASYNC0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	DP_FGYP_ASYNC0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-129. DP partial plane Window Position async0 Flow Register (DP_FG_POS_ASYNC0)

Table 42-129. DP_FG_POS_ASYNC0 Field Descriptions

Field	Description
31–27	Reserved
26–16 DP_FGXP_ASYNC0	FGXP partial plane Window X Position. partial plane Window X Position - Specifies the number of pixels between the start of full plane window X position and the beginning of the first data of new line.

Table 42-129. DP_FG_POS_ASYNC0 Field Descriptions (continued)

Field	Description
15–11	Reserved
10–0 DP_FGYP_ASYNC0	FGYP partial plane Window Y Position partial plane Window Y Position - Specifies the number of lines between the start of full plane windows Y position and the beginning of the first data.

42.2.3.4.15 DP Cursor Position and Size async0 Flow Register (DP_CUR_POS_ASYNC0)

The LCD Cursor Position Register is used to determine the starting position of the cursor relative to BG windows position.

Address 0xBASE+0xF04006C (DP_CUR_POS_ASYNC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CXW_ASYNC0					DP_CXP_ASYNC0										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CYH_ASYNC0					DP_CYP_ASYNC0										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-130. DP Cursor Position and Size async0 Flow Register (DP_CUR_POS_ASYNC0)

Table 42-130. DP_CUR_POS_ASYNC0 Field Descriptions

Field	Description
31–27 DP_CYP_ASYNC0	CYP - Cursor Y Position Represents the cursors vertical starting position Y in pixel count (from 0 to CYH).Live View Resolution Mode.
26–16 DP_CYH_ASYNC0	CYH - Cursor Height Specifies the height of the hardware cursor in pixels.
15–11 DP_CXP_ASYNC0	CXP - Cursor X Position Represents the cursors horizontal starting position X in pixel count (from 0 to CXW).
10–0 DP_CXW_ASYNC0	CXW - Cursor Width. Specifies the width of the hardware cursor in pixels.

42.2.3.4.16 DP Color Cursor Mapping async0 Flow Register (DP_CUR_MAP_ASYNC0)

The LCD Color Cursor Mapping Register defines the color of the cursor in passive or TFT color modes.

Address 0xBASE+0xF040070 (DP_CUR_MAP_ASYNC0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	CUR_COL_R_ASYNC0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CUR_COL_G_ASYNC0								CUR_COL_B_ASYNC0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-131. DP Color Cursor Mapping async0 Flow Register (DP_CUR_MAP_ASYNC0)

Table 42-131. DP_CUR_POS_ASYNC0 Field Descriptions

Field	Description
23–16 DP_CUR_COL_B_ASYNC0	CUR_COL_B - Cursor Blue Field Defines the Blue component of the cursor color in color mode 00000000No Blue. .... 11111111Full Blue.
15–8 DP_CUR_COL_G_ASYNC0	CUR_COL_G - Cursor Green Field Defines the Green component of the cursor color in color mode 00000000No Green. ..... 11111111Full Green.
7–0 DP_CUR_COL_B_ASYNC0	CUR_COL_B - Cursor Red Field Defines the Red component of the cursor color in color mode 00000000No Red. ..... 11111111Full Red.

42.2.3.4.17 DP Gamma Constants async0 Flow Register i (DP_GAMMA_C_ASYNC0_i)

This registers contains CONSTANTi parameters used for gamma correction inside the display processor (DP).

Address 0xBASE+0xF040074 (DP_GAMMA_C_ASYNC0_<i>) Access: User read/write
offset 4
range 0 .. 7

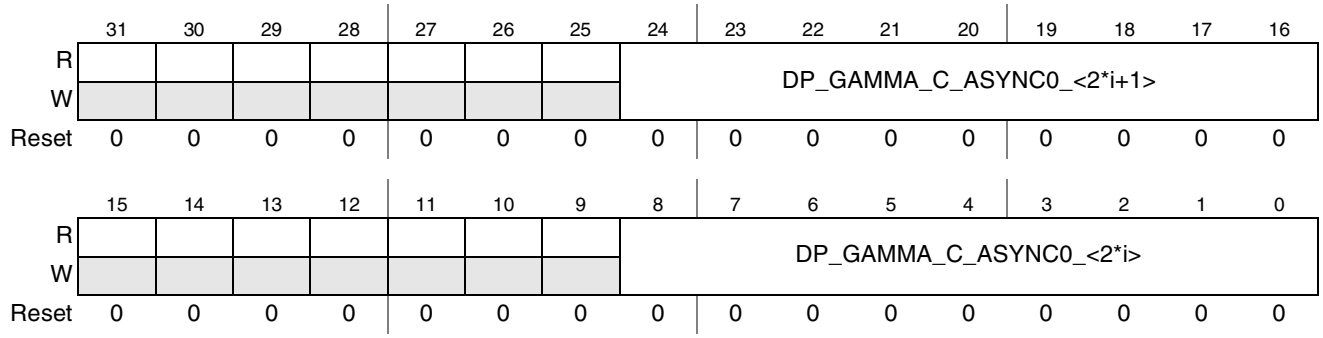


Figure 42-132. DP Gamma Constants async0 Flow Register i (DP_GAMMA_C_ASYNC0_i)

Table 42-132. DP_GAMMA_C_ASYNC0_i Field Descriptions

Field	Description
31-28	Reserved.
27-16	CONSTANTi+1 parameter of Gamma Correction.
15-9	Reserved.
8-0	CONSTANTi parameter of Gamma Correction.

42.2.3.4.18 DP Gamma Correction Slope async0 Flow Register i (DP_GAMMA_S_ASYNC0_i)

This registers contains SLOPEi parameters used for Gamma Correction inside the display processor (DP).

Address 0xBASE+0xF040094 (DP_GAMMA_S_ASYNC0_<i>) Access: User read/write
offset 4
range 0 .. 3

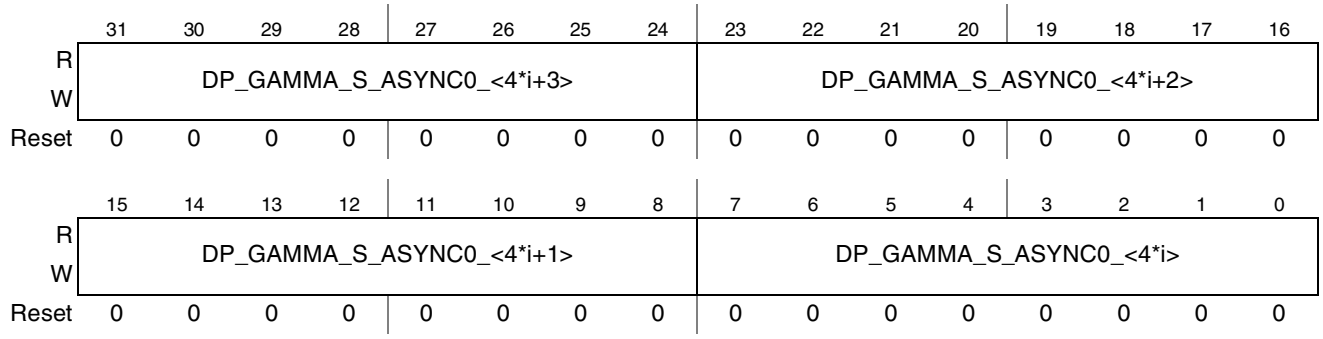


Figure 42-133. DP Gamma Correction Slope async0 Flow Register i (DP_GAMMA_S_ASYNC0_i)

Table 42-133. DP_GAMMA_S_ASYNC0_i Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Gamma Correction.
23-16	SLOPE<4*i+2> parameter of Gamma Correction.
15-8	SLOPE<4*i+1> parameter of Gamma Correction.
7-0	SLOPE<4*i> parameter of Gamma Correction.

42.2.3.4.19 DP Color Space Conversion Control async0 Flow registers (DP_CSCA_ASYNC0_i).

Address 0xBASE+0xF0400A4 (DP_CSCA_ASYNC0_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R							DP_CSC_A_ASYNC0_<2*i+1>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A_ASYNC0_<2*i>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-134. DP Color Space Conversion Control async0 Flow registers (DP_CSCA_ASYNC0_i).

Table 42-134. DP_CSCA_ASYNC0_i Field Descriptions

Field	Description
31-26	Reserved.
25-16 DP_CSC_A_ASYNC0<2*i+1>	A<2*i+1> parameter of color conversion
15-10	Reserved.
9-0 DP_CSC_A_ASYNC0<2*i>	A<2*i> parameter of color conversion.

42.2.3.4.20 DP Color Conversion Control async0 Flow registers (DP_CSC_ASYNC0_0).

Address 0xBASE+0xF0400B4 (DP_CSC_ASYNC0_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DP_CSC_S0_ASYNC0				DP_CSC_B0_ASYNC0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A8_ASYNC0									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-135. DP Color Conversion Control async0 Flow registers (DP_CSC_ASYNC0_0).

Table 42-135. DP_CSC_ASYNC0_0 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-10	Reserved.
9-0	A9 parameter of color conversion.

42.2.3.4.21 DP Color Conversion Control async0 Flow registers (DP_CSC_ASYNC0_1).

Address 0xBASE+0xF0400B8 (DP_CSC_ASYNC0_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CSC_S2_ASYNC0				DP_CSC_B2_ASYNC0											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CSC_S1_ASYNC0				DP_CSC_B1_ASYNC0											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-136. DP Color Conversion Control async0 Flow registers (DP_CSC_ASYNC0_1).

Table 42-136. DP_CSC_ASYNC0_1 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-14	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
13-0	B0 parameter of color conversion.

42.2.3.4.22 DP Common Configuration async1 Flow Register (DP_COM_CONF_ASYNC1)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF0400BC (DP_COM_CONF_ASYNC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	DP_GAMMA_YUV_EN_ASYNC1	DP_GAMMA_EN_ASYNC1	DP_CSC_YUV_SAT_MODE_ASYNC1	DP_CSC_GAMUT_SAT_EN_ASYNC1	DP_CSC_DEF_ASYNC1		0	DP_COC_ASYNC1			DP_GWCKE_ASYNC1	DP_GWAM_ASYNC1	DP_GWSEL_ASYNC1	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-137. DP Common Configuration async1 Flow Register (DP_COM_CONF_ASYNC1)

Table 42-137. DP_COM_CONF_ASYNC1 Field Descriptions

Field	Description
31–14	Reserved
13 DP_GAMMA_YUV_EN_ASYNC1	GAMMA's YUV mode enable for async flow 1 0 YUV mode is OFF. 1 YUV mode is ON.
12 DP_GAMMA_EN_ASYNC1	GAMMA_EN - Gamma correction module enable bit 0 disable 1 enable
11 DP_CSC_YUV_SAT_MODE_ASYNC1	CSC_YUV_SAT_MODE YUV saturation mode for color space conversion 0 - Y/U/V range 0 -255 1- Y range 16-235, U/V range 16-240
10 DP_CSC_GAMUT_SAT_EN_ASYNC1	CSC_GAMUT_SAT_EN Indicate if GAMUT saturation is enabled 0 - disable GAMUT mapping 1 - enable GAMUT mapping

Table 42-137. DP_COM_CONF_ASYNC1 Field Descriptions (continued)

Field	Description
9–8 DP_CSC_DEF_ASYNC1	CSC_DEF Enable or disable Color Space Conversion. 00 CSC disable 01 CSC enable after combining 10 CSC enable before combining on BG channel 11 CSC enable before combining on FG channel
6–4 DP_COC_ASYNC1	COC - Cursor Operation Control Controls the format of the cursor and the type of arithmetic operations 000Transparent, cursor is disabled. 001Full cursor. 010Reversed cursor. 011AND between full plane and cursor. 100Reserved 101OR between full plane and cursor. 110XOR between full plane and cursor. 111Reserved
3 DP_GWCKE_ASYNC1	GWCKE - Graphic Window Color Keying Enable Enable or disable graphic window color keying. 1 Enable color keying of graphic window 0 Disable color keying of graphic window
2 DP_GWAM_ASYNC1	GWAM - Graphic Window Alpha Mode Select the use of Alpha to be global or local. 1 Global Alpha. 0 Local Alpha.
1 DP_GWSEL_ASYNC1	GWSEL - Graphic Window Select Select graphic window to be on partial plane or full plane. 1 Graphic window is partial plane. 0 Graphic window is full plane.
0	Reserved

42.2.3.4.23 DP Graphic Window Control async1 Flow Register (DP_GRAPH_WIND_CTRL_ASYNC1)

This register contain common configuration parameter for the DP

Address 0xBASE+0xF0400C0 (DP_GRAPH_WIND_CTRL_ASYNC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_GWAV_ASYNC1								DP_GWCKR_ASYNC1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_GWCKG_ASYNC1								DP_GWCKB_ASYNC1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-138. DP Graphic Window Control async1 Flow Register (DP_GRAPH_WIND_CTRL_ASYNC1)

Table 42-138. DP_GRAPH_WIND_CTRL_ASYNC1 Field Descriptions

Field	Description
31–23 DP_GWAV_ASYNC1	GWAV - Graphic Window Alpha Value Defines the alpha value of graphic window used for alpha blending between graphic window and full plane. The Value the number that writing to GWAV register. The Actual Value the number, that insert to calculation in Combining Module. If Value = < 0.5- 1/256 (01111111) then Actual Value = Value. If Value >= 0.5 (10000000), then Actual Value = Value + 1/256. 00000000 Actual value is 00000000; Graphic window totally opaque i.e. overlay on LCD screen ..... 01111111Actual value is 01111111; 10000000Actual value is 10000001 10000001Actual value is 10000010 ..... 11111110Actual value is 11111111 11111111Actual value is 100000000;Graphic window totally transparent i.e. not displayed on LCD screen
31–23 DP_GWCKR_ASYNC1	GWCKR - Graphic Window Color Keying Red Component Defines the red component of graphic window color keying. 00000000No red ..... 11111111Full red

Table 42-138. DP_GRAPH_WIND_CTRL_ASYNC1 Field Descriptions (continued)

Field	Description
31–23 DP_GWCKG_ASYNC1	GWCKG - Graphic Window Color Keying Green Component Defines the green component of graphic window color keying. 00000000No Green ..... 11111111Full Green
31–23 DP_GWCKB_ASYNC1	GWCKB - Graphic Window Color Keying Blue Component Defines the blue component of graphic window color keying. 00000000No blue ..... 11111111Full blue

42.2.3.4.24 DP partial plane Window Position async1 Flow Register (DP_FG_POS_ASYNC1)

The DP partial plane Window Position Register is used to determine the starting position of the partial plane window relative to BG window position.

This Register is used for the synchronous flow only.

Address 0xBASE+0xF0400C4 (DP_FG_POS_ASYNC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	DP_FGXP_ASYNC1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	DP_FGYP_ASYNC1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-139. DP partial plane Window Position async1 Flow Register (DP_FG_POS_ASYNC1)

Table 42-139. DP_FG_POS_ASYNC1 Field Descriptions

Field	Description
31–27	Reserved
26–16 DP_FGXP_ASYNC1	FGXP partial plane Window X Position. partial plane Window X Position - Specifies the number of pixels between the start of full plane window X position and the beginning of the first data of new line.

Table 42-139. DP_FG_POS_ASYNC1 Field Descriptions (continued)

Field	Description
15–11	Reserved
10–0 DP_FGYP_ASYNC1	FGYP partial plane Window Y Position partial plane Window Y Position - Specifies the number of lines between the start of full plane windows Y position and the beginning of the first data.

42.2.3.4.25 DP Cursor Position and Size async1 Flow Register (DP_CUR_POS_ASYNC1)

The LCD Cursor Position Register is used to determine the starting position of the cursor relative to BG windows position.

Address 0xBASE+0xF0400C8 (DP_CUR_POS_ASYNC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CXW_ASYNC1								DP_CXP_ASYNC1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CYH_ASYNC1								DP_CYP_ASYNC1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-140. DP Cursor Position and Size async1 Flow Register (DP_CUR_POS_ASYNC1)

Table 42-140. DP_CUR_POS_ASYNC1 Field Descriptions

Field	Description
31–27 DP_CYP_ASYNC1	CYP - Cursor Y Position Represents the cursors vertical starting position Y in pixel count (from 0 to CYH).Live View Resolution Mode.
26–16 DP_CYH_ASYNC1	CYH - Cursor Height Specifies the height of the hardware cursor in pixels.
15–11 DP_CXP_ASYNC1	CXP - Cursor X Position Represents the cursors horizontal starting position X in pixel count (from 0 to CXW).
10–0 DP_CXW_ASYNC1	CXW - Cursor Width. Specifies the width of the hardware cursor in pixels.

42.2.3.4.26 DP Color Cursor Mapping async1 Flow Register (DP_CUR_MAP_ASYNC1)

The LCD Color Cursor Mapping Register defines the color of the cursor in passive or TFT color modes.

Address 0xBASE+0xF0400CC (DP_CUR_MAP_ASYNC1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	CUR_COL_R_ASYNC1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CUR_COL_G_ASYNC1								CUR_COL_B_ASYNC1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-141. DP Color Cursor Mapping async1 Flow Register (DP_CUR_MAP_ASYNC1)

Table 42-141. DP_CUR_POS_ASYNC1 Field Descriptions

Field	Description
23–16 DP_CUR_COL_B_ASYNC1	CUR_COL_B - Cursor Blue Field Defines the Blue component of the cursor color in color mode 00000000No Blue. .... 11111111Full Blue.
15–8 DP_CUR_COL_G_ASYNC1	CUR_COL_G - Cursor Green Field Defines the Green component of the cursor color in color mode 00000000No Green. ..... 11111111Full Green.
7–0 DP_CUR_COL_B_ASYNC1	CUR_COL_B - Cursor Red Field Defines the Red component of the cursor color in color mode 00000000No Red. ..... 11111111Full Red.

42.2.3.4.27 DP Gamma Constants async1 Flow Register i (DP_GAMMA_C_ASYNC1_i)

This registers contains CONSTANTi parameters used for gamma correction inside the display processor (DP).

Address 0xBASE+0xF0400D0 (DP_GAMMA_C_ASYNC1_<i>) Access: User read/write
offset 4
range 0 .. 7

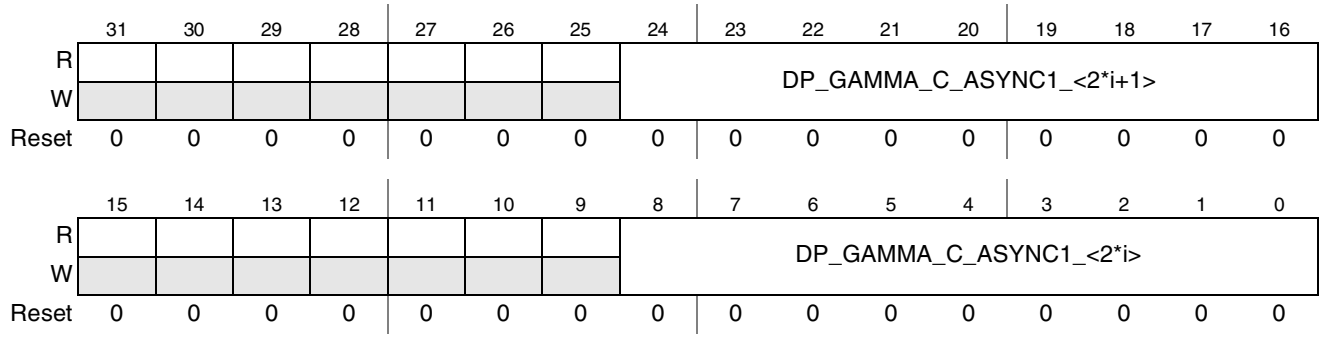


Figure 42-142. DP Gamma Constants async1 Flow Register i (DP_GAMMA_C_ASYNC1_i)

Table 42-142. DP_GAMMA_C_ASYNC1_i Field Descriptions

Field	Description
31-28	Reserved.
27-16	CONSTANTi+1 parameter of Gamma Correction.
15-9	Reserved.
8-0	CONSTANTi parameter of Gamma Correction.

42.2.3.4.28 DP Gamma Correction Slope async1 Flow Register i (DP_GAMMA_S_ASYNC1_i)

This registers contains SLOPEi parameters used for Gamma Correction inside the display processor (DP).

Address 0xBASE+0xF0400F0 (DP_GAMMA_S_ASYNC1_<i>) Access: User read/write
offset 4
range 0 .. 3

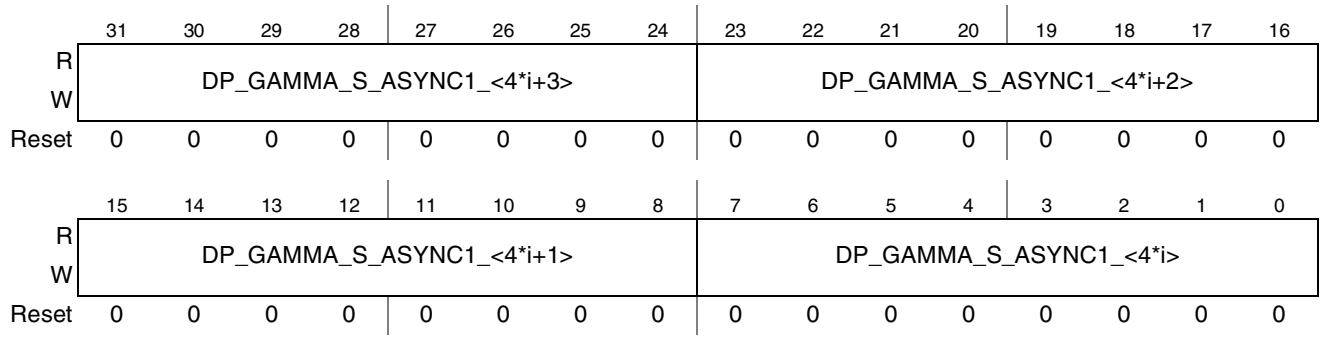


Figure 42-143. DP Gamma Correction Slope async1 Flow Register i (DP_GAMMA_S_ASYNC1_i)

Table 42-143. DP_GAMMA_S_ASYNC1_i Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Gamma Correction.
23-16	SLOPE<4*i+2> parameter of Gamma Correction.
15-8	SLOPE<4*i+1> parameter of Gamma Correction.
7-0	SLOPE<4*i> parameter of Gamma Correction.

42.2.3.4.29 DP Color Space Conversion Control async1 Flow registers (DP_CSCA_ASYNC1_i).

Address 0xBASE+0xF040100 (DP_CSCA_ASYNC1_<i>)
offset 4
range 0 .. 3

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R							DP_CSC_A_ASYNC1_<2*i+1>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A_ASYNC1_<2*i>									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-144. DP Color Conversion Control async1 Flow registers (DP_CSC_ASYNC1_0).

Table 42-144. DP_CSCA_ASYNC1_i Field Descriptions

Field	Description
31-26	Reserved.
25-16 DP_CSC_A_ASYNC1<2*i+1>	A<2*i+1> parameter of color conversion
15-10	Reserved.
9-0 DP_CSC_A_ASYNC1<2*i>	A<2*i> parameter of color conversion.

42.2.3.4.30 DP Color Conversion Control async1 Flow registers (DP_CSC_ASYNC1_0).

Address 0xBASE+0xF040110 (DP_CSC_ASYNC1_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DP_CSC_S0_ASYNC1				DP_CSC_B0_ASYNC1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R							DP_CSC_A8_ASYNC1									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-145. DP Color Conversion Control Sync Flow registers (DP_CSC_SYNC_0).

Table 42-145. DP_CSC_ASYNC1_0 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-10	Reserved.
9-0	A9 parameter of color conversion.

42.2.3.4.31 DP Color Conversion Control async1 Flow registers (DP_CSC_ASYNC1_1).

Address 0xBASE+0xF040114 (DP_CSC_ASYNC1_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DP_CSC_S2_ASYNC1				DP_CSC_B2_ASYNC1											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DP_CSC_S1_ASYNC1				DP_CSC_B1_ASYNC1											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-146. DP Color Conversion Control async1 Flow registers (DP_CSC_ASYNC1_1).

Table 42-146. DP_CSC_ASYNC1_1 Field Descriptions

Field	Description
31-30	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
29-16	B0 parameter of color conversion.
15-14	S0 parameter of color conversion. 00 scale factor of 2 01 scale factor of 1 10 scale factor of 0 11 scale factor of -1
13-0	B0 parameter of color conversion.

42.2.3.4.32 DP Debug Control register (DP_DEBUG_CNT).

This is the debug unit control register. This register is not stored in the SRM.

Address 0xBASE+0xE0180BC (DP_DEBUG_CNT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	BRAKE_CNT_1			BRAKE_STATUS_EN_1	BRAKE_CNT_0			BRAKE_STATUS_EN_0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-147. DP Debug Control register (DP_DEBUG_CNT).

Table 42-147. DP_DEBUG_CNT Field Descriptions

Field	Description
7-5 BRAKE_CNT_1	The async flow can be broken multiple times. It possible to control which breaking event will cause the interrupt. This field counts the breaking events for unit #1
4 BRAKE_STATUS_EN_1	This bit enables the break/status unit #1
3-1 BRAKE_CNT_0	The async flow can be broken multiple times. It possible to control which breaking event will cause the interrupt. This field counts the breaking events for unit #0
0 BRAKE_STATUS_EN_0	This bit enables the break/status unit #0

42.2.3.4.33 DP Debug Status register (DP_DEBUG_STAT).

Address 0xBASE+0xE0180C0 (DP_DEBUG_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CYP_EN_OLD_1	COMBYP_EN_OLD_1	FG_ACTIVE_1	V_CNT_OLD_1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	CYP_EN_OLD_0	COMBYP_EN_OLD_0	FG_ACTIVE_0	V_CNT_OLD_0										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-148. DP Debug Status register (DP_DEBUG_STAT).

Table 42-148. DP_DEBUG Field Descriptions

Field	Description
29 CYP_EN_OLD_1	The async flow has been broken in the middle of a cursor (This filed is relevant for debug unit #1)
28 COMBYP_EN_OLD_1	the async1 flow has been broken in the middle of combining (This filed is relevant for debug unit #1)
27 FG_ACTIVE_1	Displaying the partial frame has been started (This filed is relevant for debug unit #1)
26-16 V_CNT_OLD_1	The exact row where the async flow has been broken (This filed is relevant for debug unit #0)
13 CYP_EN_OLD_0	The async flow has been broken in the middle of a cursor (This filed is relevant for debug unit #0)
12 COMBYP_EN_OLD_0	the async flow has been broken in the middle of combining (This filed is relevant for debug unit #0)
11 FG_ACTIVE_0	Displaying the partial frame has been started for async flow (This filed is relevant for debug unit #0)
10-0 V_CNT_OLD_0	The exact row where the async flow has been broken (This filed is relevant for debug unit #0)

42.2.3.5 IC Registers

42.2.3.5.1 IC Configuration Register (IC_CONF)

This register contains control parameter for IC 3 tasks (pre-processing for encoding, pre-processing for view-finder and post processing).

Address 0xBASE+0xE020000 (IC_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI_MEM_WR_EN	RWS_EN	IC_KEY_COLOR_EN	IC_GLB_LOC_A	0	0	0	0	0	0	0	PP_ROT_EN	PP_CMB	PP_CSC2	PP_CSC1	PP_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	PRPVF_ROT_EN	PRPVF_CMB	PRPVF_CSC2	PRPVF_CSC1	PRPVF_EN	0	0	0	0	0	PRPENC_ROT_EN	PRPENC_CSC1	PRPENC_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-149. IC Configuration Register (IC_CONF)

Table 42-149. IC_CONF Field Descriptions

Field	Description
31 CSI_MEM_WR_EN	CSI direct memory write enable. This bit enables writing data from sensor directly to memory even when a raw sensor is not attached. 0 CSI direct writing to memory is disabled. 1 CSI direct writing to memory is enabled.
30 RWS_EN	Raw sensor enable. This bit indicate if a raw sensor is attached (Bayer format). 0 Raw sensor is not attached. 1 Raw sensor is attached. This bit is used together with the CSI_MEM_WR_EN bit as follows: CSI_MEM_WR_EN=0, RWS_EN=0 - data is fed from the CSI to the IC for processing; CSI_MEM_WR_EN=1, RWS_EN=0 - data is fed from the CSI to the IC for processing and also for writing to the system memory; CSI_MEM_WR_EN=0, RWS_EN=1 - data is fed from the CSI to the system memory (via the IC) and from the system memory to the IC for processing; CSI_MEM_WR_EN=1, RWS_EN=1 - non-valid configuration.
29 IC_KEY_COLOR_EN	Key color enable. This bit enables the key color feature. 0 Key color is disabled. 1 Key color is enabled.
28 IC_GLB_LOC_A	Global alpha. This bit select the source of alpha parameter. 0 Alpha parameter is local. 1 Alpha parameter is global.
27–21	Reserved
20 PP_ROT_EN	Post-processing rotation task enable. This bit enable post-processing rotation task. 0 Rotation is disabled. 1 Rotation is enabled.
19 PP_CMB	Post-processing task combining enable. This bit enables combining. 0 Combining is disabled. 1 Combining is enabled.
18 PP_CSC2	
17 PP_CSC1	Post-processing task color conversion YUV-->RGB enable. This bit enables YUV-->RGB. 0 YUV-->RGB is disabled. 1 YUV-->RGB is enabled.
16 PP_EN	Post-processing task enable. This bit enables the post-processing task. 0 Task is disabled. 1 Task is enabled.
15–13	Reserved

Table 42-149. IC_CONF Field Descriptions (continued)

Field	Description
12 PRPVF_ROT_EN	Preprocessing rotation task for viewfinder enable. This bit enable preprocessing rotation task for viewfinder. 0 Rotation is disabled. 1 Rotation is enabled.
11 PRPVF_CMB	Preprocessing task for view-finder combining enable. This bit enables combining. 0 Combining is disabled. 1 Combining is enabled.
10 PRPVF_CSC2	Pre-processing task for view-finder second color conversion enable. This bit enables second color conversion. 0 Second color conversion is disabled. 1 Second color conversion is enabled.
9 PRPVF_CSC1	Pre-processing task for view-finder first color conversion enable. This bit enables first color conversion. 0 First color conversion is disabled. 1 First color conversion is enabled.
8 PRPVF_EN	Preprocessing task for view-finder enable. This bit enables the view-finder task. 0 Task is disabled. 1 Task is enabled.
7–3	Reserved
2 PRPENC_ROT_EN	Preprocessing rotation task for encoding enable. This bit enable preprocessing rotation task for encoding. 0 Rotation is disabled. 1 Rotation is enabled.
1 PRPENC_CSC1	Preprocessing task for encoding color conversion enable. This bit enables color conversion. 0 Color conversion is disabled. 1 Color conversion is enabled.
0 PRPENC_EN	Preprocessing task for encoding enable. This bit enables the encoding task. 0 Task is disabled. 1 Task is enabled.

42.2.3.5.2 IC Preprocessing Encoder Resizing Coefficients Register (IC_PRP_ENC_RSC)

This register contains the resizing and downsizing parameters for preprocessing task for encoding.

Address 0xBASE+0xE020004 (IC_PRP_ENC_RSC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRPENC_DS_R_V		PRPENC_RS_R_V													
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRPENC_DS_R_H		PRPENC_RS_R_H													
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-150. IC Preprocessing Encoder Resizing Coefficients Register (IC_PRP_ENC_RSC)

Table 42-150. IC_PRP_ENC_RSC Field Descriptions

Field	Description
31–30 PRPENC_DS_R_V	Preprocessing task for encoding downsizing vertical ratio. This field contains the downsizing vertical coefficient of preprocessing for encoding.
29–16 PRPENC_RS_R_V	Preprocessing task for encoding resizing vertical ratio. This field contains the resizing vertical coefficient of preprocessing for encoding. Resizing ratio is equal to PRPENC_RS_R_V: M Where M = 2^13; SI - input size; SO - output size PRPENC_RS_R_V = floor(M*(SI-1)/(SO-1));
15–14 PRPENC_DS_R_H	Preprocessing task for encoding downsizing horizontal ratio. This field contains the downsizing horizontal coefficient of preprocessing for encoding. Values: 00 1 01 2 10 4 11 RSV
13–0 PRPENC_RS_R_H	Preprocessing task for encoding resizing horizontal ratio. This field contains the resizing horizontal coefficient of preprocessing for encoding. Resizing ratio is equal to PRPENC_RS_R_H: M Where M = 2^13; SI - input size; SO - output size PRPENC_RS_R_H = floor(M*(SI-1)/(SO-1));

42.2.3.5.3 IC Preprocessing View-Finder Resizing Coefficients Register (IC_PRP_VF_RSC)

This register contains the resizing and downsizing parameters for preprocessing task for display.

Address 0xBASE+0xE020008 (IC_PRP_VF_RSC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRPVF_DS_R_V		PRPVF_RS_R_V													
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PRPVF_DS_R_H		PRPVF_RS_R_H													
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-151. IC Preprocessing View-Finder Resizing Coefficients Register (IC_PRP_VF_RSC)

Table 42-151. IC_PRP_VF_RSC Field Descriptions

Field	Description
31–30 PRPVF_DS_R_V	Preprocessing task for encoding downsizing vertical ratio. This field contains the downsizing vertical coefficient of preprocessing for view-finder.
29–16 PRPVF_RS_R_V	Preprocessing task for encoding resizing vertical ratio. This field contains the resizing vertical coefficient of preprocessing for view-finder. Resizing ratio is equal to PRPVF_RS_R_V: M Where M = 2^13; SI - input size; SO - output size PRPVF_RS_R_V = floor(M*(SI-1)/(SO-1));
15–14 PRPVF_DS_R_H	Preprocessing task for encoding downsizing horizontal ratio. This field contains the downsizing horizontal coefficient of preprocessing for view-finder. Values: 00 1 01 2 10 4 11 RSV
13–0 PRPVF_RS_R_H	Preprocessing task for view-finding resizing horizontal ratio. This field contains the resizing horizontal coefficient of preprocessing task for view-finder. Resizing ratio is equal to PRPVF_RS_R_H: M Where M = 2^13; SI - input size; SO - output size PRPVF_RS_R_H = floor(M*(SI-1)/(SO-1));

42.2.3.5.4 IC Post-Processing Resizing Coefficients Register (IC_PP_RSC)

This register contains the resizing and downsizing parameters for post-processing task for display.

Address 0xBASE+0xE02000C (IC_PP_RSC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PP_DS_R_V				PP_RS_R_V											
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PP_DS_R_H				PP_RS_R_H											
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-152. IC Post-Processing Resizing Coefficients Register (IC_PP_RSC)

Table 42-152. IC_PP_RSC Field Descriptions

Field	Description
31–30 PP_DS_R_V	Post-processing task downsizing vertical ratio. This field contains the downsizing vertical coefficient of post-processing.
29–16 PP_RS_R_V	Post-processing task resizing vertical ratio. This field contains the resizing vertical coefficient of post-processing. Resizing ratio is equal to PP_RS_R_V: M Where M = 2^13; SI - input size; SO - output size PP_RS_R_V = floor(M*(SI-1)/(SO-1));
15–14 PP_DS_R_H	Post-processing task downsizing horizontal ratio. This field contains the downsizing horizontal coefficient of post-processing. 00 1 01 2 10 4 11 RSV
13–0 PP_RS_R_H	Post-processing task resizing horizontal ratio. This field contains the resizing horizontal coefficient of post-processing. Resizing ratio is equal to PP_RS_R_H: M Where M = 2^13; SI - input size; SO - output size PP_RS_R_H = floor(M*(SI-1)/(SO-1));

42.2.3.5.5 IC Combining Parameters Register 1 (IC_CMBP_1)

Address 0xBASE+0xE020010 (IC_CMBP_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IC_PP_ALPHA_V								IC_PRPVF_ALPHA_V							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-153. IC Combining Parameters Register 1 (IC_CMBP_1)

Table 42-153. IC_CMBP_1 Field Descriptions

Field	Description
31–16	Reserved
15–8 IC_PP_ALPHA_V	Post-processing task global alpha. This field contains the global alpha value of post-processing
7–0 IC_PRPVF_ALPHA_V	Preprocessing task for viewfinder global alpha. This field contains the global alpha value of preprocessing for viewfinder.

42.2.3.5.6 IC Combining Parameters Register 2 (IC_CMBP_2)

Address 0xBASE+0xE020014 (IC_CMBP_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	IC_KEY_COLOR_R							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IC_KEY_COLOR_G								IC_KEY_COLOR_B							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-154. IC Combining Parameters Register 2 (IC_CMBP_2)

Table 42-154. IC_CMBP_2 Field Descriptions

Field	Description
31–24	Reserved
23–16 IC_KEY_COLOR_R	Key color red.
15–8 IC_KEY_COLOR_G	Key color green.
7–0 IC_KEY_COLOR_B	Key color blue.

42.2.3.5.7 IC IDMAC Parameters 1 Register (IC_IDMAC_1)

This registers contains information about the

Address 0xBASE+0xE020018 (IC_IDMAC_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	ALT_CB7_BURST_16	ALT_CB6_BURST_16	0	T3_FLIP_RS	T2_FLIP_RS	T1_FLIP_RS	T3_FLIP_UD	T3_FLIP_LR	T3_ROT	T2_FLIP_UD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T2_FLIP_LR	T2_ROT	T1_FLIP_UD	T1_FLIP_LR	T1_ROT	0	0	0	CB7_BURST_16	CB6_BURST_16	CB5_BURST_16	CB4_BURST_16	CB3_BURST_16	CB2_BURST_16	CB1_BURST_16	CB0_BURST_16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-155. IC IDMAC Parameters 1 Register (IC_IDMAC_1)

Table 42-155. IC_IDMAC_1 Field Descriptions

Field	Description
31–26	Reserved
25 ALT_CB7_BURST_16	This bit defines the number of pixels within a burst (burst size) coming from the IDMAC for IC's CB7 when used in alternate mode. For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
24 ALT_CB6_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB6 when used in alternate mode. For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
23	Reserved
22 T3_FLIP_RS	LEFT/RIGHT flip for Post Processing (PP) task; his bit affect the flipping done on the resizing unit. The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip
21 T2_FLIP_RS	LEFT/RIGHT flip for View Finder (VF) task; this bit affect the flipping done on the resizing unit. The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip
20 T1_FLIP_RS	LEFT/RIGHT flip for Encoding (ENC) task; this bit affect the flipping done on the resizing unit. The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip
19 T3_FLIP_UD	UP/DOWN flip for Post Processing (PP) task The value of this field must be identical to the corresponding channel's VF parameters in the IDMAC's CPMEM 1 - Vertical flip enable 0 - no flip
18 T3_FLIP_LR	LEFT/RIGHT flip for Post Processing (PP) task; this bit affect the flipping done on the rotation unit The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip

Table 42-155. IC_IDMAC_1 Field Descriptions (continued)

Field	Description
17 T3_ROT	Rotation for Post Processing (PP) task The value of this field must be identical to the corresponding channel's ROT parameters in the IDMAC's CPMEM 1 - 90 degree rotation clockwise 0 - no rotation
16 T2_FLIP_UD	UP/DOWN flip for View Finder (VF) task The value of this field must be identical to the corresponding channel's VF parameters in the IDMAC's CPMEM 1 - Vertical flip enable 0 - no flip
15 T2_FLIP_LR	LEFT/RIGHT flip for View Finder (VF) task; this bit affect the flipping done on the rotation unit The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip
14 T2_ROT	Rotation for View Finder (VF) task The value of this field must be identical to the corresponding channel's ROT parameters in the IDMAC's CPMEM 1 - 90 degree rotation clockwise 0 - no rotation
13 T1_FLIP_UD	UP/DOWN flip for Encoding (ENC) task The value of this field must be identical to the corresponding channel's VF parameters in the IDMAC's CPMEM 1 - Vertical flip enable 0 - no flip
12 T1_FLIP_LR	LEFT/RIGHT flip for Encoding (ENC) task; this bit affect the flipping done on the rotation unit The value of this field must be identical to the corresponding channel's HF parameter in the IDMAC's CPMEM 1 - horizontal flip enabled 0 - no flip
11 T1_ROT	Rotation for Encoding (ENC) task The value of this field must be identical to the corresponding channel's ROT parameters in the IDMAC's CPMEM 1 - 90 degree rotation clockwise 0 - no rotation
10–8	Reserved
7 CB7_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB7 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111

Table 42-155. IC_IDMAC_1 Field Descriptions (continued)

Field	Description
6 CB6_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB6 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
5 CB5_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB5 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
4 CB4_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB4 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
3 CB3_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB3 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
2 CB2_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB2 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
1 CB1_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB1 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111
0 CB0_BURST_16	This bit defines the number of active cycles within a burst (burst size) coming from the IDMAC for IC's CB0 For pixel data the number of pixels should match the NPB[6:2] value on the IDMAC's CPMEM 0 - Burst size is 8 pixels; The Matching NPB[6:2] should be 00111 1 - Burst size is 16 pixels; The Matching NPB[6:2] should be 01111

42.2.3.5.8 IC IDMAC Parameters 2 Register (IC_IDMAC_2)

Address 0xBASE+0xE02001C (IC_IDMAC_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	T3_FR_HEIGHT										T2_FR_HEIGHT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T2_FR_HEIGHT						T1_FR_HEIGHT									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-156. IC IDMAC Parameters 2 Register (IC_IDMAC_2)

Table 42-156. IC_IDMAC_2 Field Descriptions

Field	Description
31–24	Reserved
29–20 T3_FR_HEIGHT	Frame Height for Post Processing (PP) task The value of this field must be identical to the corresponding FH channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1
19–10 T2_FR_HEIGHT	Frame Height for View Finder (VF) task The value of this field must be identical to the corresponding FH channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1
9–0 T1_FR_HEIGHT	Frame Height for Encoding (ENC) task The value of this field must be identical to corresponding FH channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1

42.2.3.5.9 IC IDMAC Parameters 3 Register (IC_IDMAC_3)

Address 0xBASE+0xE020020 (IC_IDMAC_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	T3_FR_WIDTH										T2_FR_WIDTH			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T2_FR_WIDTH						T1_FR_WIDTH									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-157. IC IDMAC Parameters 3 Register (IC_IDMAC_3)

Table 42-157. IC_IDMAC_2 Field Descriptions

Field	Description
31–24	Reserved
29–20 T3_FR_WIDTH	Frame Width for Post Processing (PP) task The value of this field must be identical to the corresponding FW channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1
19–10 T2_FR_WIDTH	Frame Width for View Finder (VF) task The value of this field must be identical to the corresponding FW channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1
9–0 T1_FR_WIDTH	Frame Width for Encoding (ENC) task The value of this field must be identical to corresponding FW channel's parameters in the IDMAC's CPMEM. This parameter refer's to the output's size -1

42.2.3.5.10 IC IDMAC Parameters 4 Register (IC_IDMAC_4)

Address 0xBASE+0xE020024 (IC_IDMAC_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	rm_brdg_max_rq				ibm_brdg_max_rq				mpm_dmfc_brdg_max_rq				mpm_rw_brdg_max_rq			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-158. IC IDMAC Parameters 3 Register (IC_IDMAC_3)

Table 42-158. IC_IDMAC_2 Field Descriptions

Field	Description
31–16	Reserved
15-12 rm_brdg_max_rq	RM memory Bridge Max Requests 0000 Feature is disabled 0001 Max request is 1 .... 1111 Max request are15
11-8 ibm_brdg_max_rq	IBM memory Bridge Max Requests 0000 Feature is disabled 0001 Max request is 1 .... 1111 Max request are15
7-4 mpm_dmfc_brdg_max_rq	MPM memory Bridge Max Requests for the IC DMFC interface 0000 Feature is disabled 0001 Max request is 1 .... 1111 Max request are15
3-0 mpm_rw_brdg_max_rq	MPM memory Bridge Max Requests between MPM's read and writes 0000 Feature is disabled 0001 Max request is 1 .... 1111 Max request are15

42.2.3.6 IRT Registers

42.2.3.7 CSI0 registers

42.2.3.7.1 CSI0 Sensor Configuration Register (CSI0_SENS_CONF)

Address 0xBASE+0xE030000 (CSI0_SENS_CONF) Access: User read-write

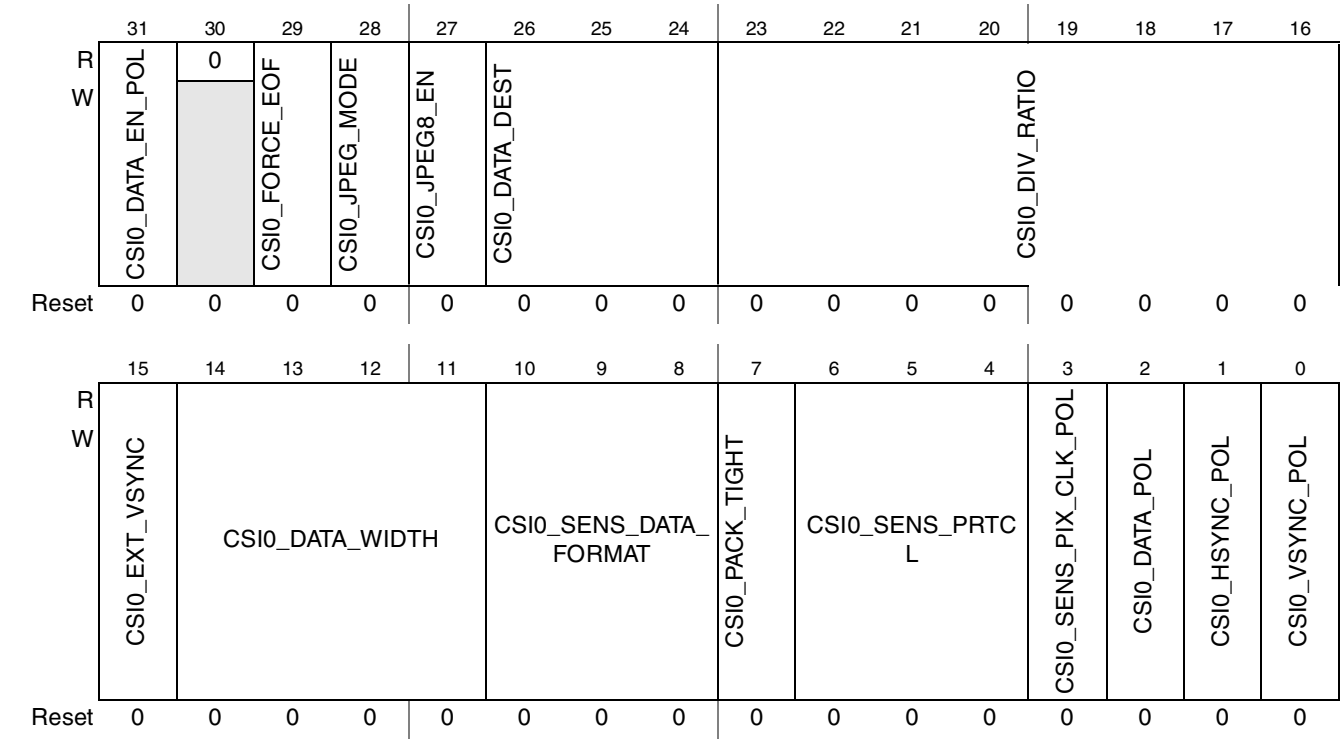


Figure 42-159. CSI0 Sensor Configuration Register (CSI0_SENS_CONF)

Table 42-159. Register Field Descriptions

31 CSI0_DATA_EN_POL	Invert IPP_IND_SENSB_DATA_EN input. This bit selects the polarity of IPP_IND_SENSB_DATA_EN signal. 0 IPP_IND_SENSB_DATA_EN is directly applied to internal circuitry. 1 IPP_IND_SENSB_DATA_EN is inverted before applied to internal circuitry.
30	Reserved, should be cleared.
29 CSI0_FORCE_EOF	Force End of frame This is a self clear bit allowing the user to force an End-of-frame event; This bit can be used in cases where the frame sent by the sensor was not completed. 1 - force end of frame 0 - no action
28 CSI0_JPEG_MODE	JPEG Mode - this bit defines the mode of the control signals when working in JPEG mode 1 - The data is valid as long as HSYNC and VSYNC signals are active; HSYNC is valid for single frame 0 - The frame starts withe the assertion of VSYNC. The frame ends on the next VSYNC or by setting the CSI0_FORCE_EOF bit

Table 42-159. Register Field Descriptions (continued)

27 CSI0_JPEG8_EN	JPEG8 enable bit 1 - JPEG8 detection is enabled 0 - JPEG8 is disabled
26–24 CSI0_DATA_DEST	These bits enable the destination of the data coming from the CSI. CSI0_DATA_DEST[0] - CSI0_DATA_DEST[1] - destination is IC CSI0_DATA_DEST[2] - destination is IDMAC via SMFC
23–16 CSI0_DIV_RATIO	DIV Ratio Clock division ratio minus 1. This field defines the division ratio of HSP_CLK into SENSB_MCLK: SENSB_MCLK rate = HSP_CLK rate /(DIV_RATIO+1)
15 CSI0_EXT_VSYNC	External VSYNC enable. This bits select between external and internal VSYNC. 0 Internal VSYNC mode. 1 External VSYNC mode.
14–11 CSI0_DATA_WIDTH	Data width. This field defines the number of bits per color. Values: 0000 - 4 bits per color 0001 - 8 bits per color 0010 - 9 bits per color 0011 - 10 bits per color 0100 - 11 bits per color 0101 - 12 bits per color 0110 - 13 bits per color 0111 - 14 bits per color 1000 - 15 bits per color 1001 - 16 bits per color
10–8 CSI0_SENS_DATA_FORMAT	Data format from the sensor. This field defines the data format for the input of the CSI sensor. Values: 000 - full RGB or YUV444 001 - YUV422 (YUYV...) 010 - YUV422 (UYVY...) 011 - Bayer or Generic data 100 - RGB565 101 - RGB555 110 - RGB444 111 - JPEG
7 CSI0_PACK_TIGHT	CSI0 Pack Tight When the data format is YUV or RGB and the component's width is 9-16 bits, it can be sent to the memory in 2 different ways 1 - Three 10 bits components are packed into a 32 bit word. Color extension/reduction is performed 0 - Each component is written as a 16 bit word where the MSB is written to bit #15, color extension is done for the remaining least significant bits.

Table 42-159. Register Field Descriptions (continued)

6–4 CSI0_SENS_PRTCL	Sensor protocol. This bit defines the sensor timing/data mode protocol. Values: 000 Gated clock mode 001 Non-gated clock mode 010 CCIR progressive mode (BT.656) 011 CCIR interlaced mode (BT.656) 100 CCIR progressive (BT.1120 DDR mode: data arrives on every edge of the clock) 101 CCIR progressive (BT.1120 SDR mode: data arrives only on the positive edge of the clock) 110 CCIR interlaced mode (BT.1120 DDR mode: data arrives on every edge of the clock) 111 CCIR interlaced mode (BT.1120 SDR mode: data arrives only on the positive edge of the clock)
3 CSI0_SENS_PIX_CLK_POL	Invert pixel clock input. This bit selects the polarity of pixel clock. 0 pixel clock is directly applied to internal circuitry. 1 pixel clock is inverted before applied to internal circuitry.
2 CSI0_DATA_POL	Invert data input. This bit selects the polarity of data input. 0 data lines are directly applied to internal circuitry. 1 data lines are inverted before applied to internal circuitry.
1 CSI0_HSYNC_POL	Invert IPP_IND_SENSB_HSYNC input. This bit selects the polarity of IPP_IND_SENSB_HSYNC signal. 0 IPP_IND_SENSB_HSYNC is directly applied to internal circuitry. 1 IPP_IND_SENSB_HSYNC is inverted before applied to internal circuitry.
0 CSI0_VSYNC_POL	Invert IPP_IND_SENSB_VSYNC input. This bit selects the polarity of IPP_IND_SENSB_VSYNC signal. 0 IPP_IND_SENSB_VSYNC is not inverted before applied to internal circuitry. 1 IPP_IND_SENSB_VSYNC is inverted before applied to internal circuitry.

42.2.3.7.2 CSI0 Sense Frame Size Register (CSI0_SENS_FRM_SIZE)

Address 0xBASE+0xE030004 (CSI0_SENS_FRM_SIZE) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	CSI0_SENS_FRM_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	CSI0_SENS_FRM_WIDTH												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-160. CSI0 Sense Frame Size Register (CSI0_SENS_FRM_SIZE)

Table 42-160. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 CSI0_SENS_FRM_HEIGHT	Sensor frame height minus 1. This field defines the sensor frame rows number minus 1.
15–13	Reserved, should be cleared.
12–0 CSI0_SENS_FRM_WIDTH	Sensor frame width minus 1. This field defines the sensor frame column number minus 1.

42.2.3.7.3 CSI0 Actual Frame Size Register (CSI0_ACT_FRM_SIZE)

Address 0xBASE+0xE030008 (CSI0_ACT_FRM_SIZE) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	CSI0_ACT_FRM_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	CSI0_ACT_FRM_WIDTH												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-161. CSI0 Actual Frame Size Register (CSI0_ACT_FRM_SIZE)

Table 42-161. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 CSI0_ACT_FRM_HEIGHT	Actual frame height minus 1. This field defines the CSI output frame rows number minus 1.
15–13	Reserved, should be cleared.
12–0 CSI0_ACT_FRM_WIDTH	Actual frame width minus 1. This field defines the CSI output frame columns number minus 1.

42.2.3.7.4 CSI0 Output Control Register (CSI0_OUT_FRM_CTRL)

Address 0xBASE+0xE03000C (CSI0_OUT_FRM_CTRL) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI0_HORZ_DWNS	CSI0_VERT_DWNS	0	CSI0_HSC												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CSI0_VSC											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-162. CSI0 Output Control Register (CSI0_OUT_FRM_CTRL)

Table 42-162. Register Field Descriptions

Field	Description
31 CSI0_HORZ_DWNS	Enable horizontal downsizing (decimation) by 2. 0 Downsizing disabled 1 Downsizing enabled
30 CSI0_VERT_DWNS	Enable vertical downsizing (decimation) by 2. 0 Downsizing disabled 1 Downsizing enabled
29	Reserved, should be cleared.
28–16 CSI0_HSC	Horizontal skip. This field defines the number of columns to skip. In Interlaced mode this number refers to the number of lines per field
15–12	Reserved, should be cleared.
11–0 CSI0_VSC	Vertical skip. This field defines the number of rows to skip.

42.2.3.7.5 CSI0 Test Control Register (CSI0_TST_CTRL)

Address 0xBASE+0xE030010 (CSI0_TST_CTRL) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_TEST_GEN_MODE	CSI0_PG_B_VALUE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI0_PG_G_VALUE								CSI0_PG_R_VALUE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-163. CSI0 Test Control Register (CSI0_TST_CTRL)

Table 42-163. Register Field Descriptions

Field	Description
31–25	Reserved, should be cleared.
2 TEST_GEN_MODE	Test generator mode. This bit activates the signal generation. 0 Test signal generator is inactive. 1 Test signal generator is active.
23–16 PG_B_VALUE	Pattern generator B value. This field selects the B value for the generated pattern of even pixel.
15–8 PG_G_VALUE	Pattern generator G value. This field selects the G value for the generated pattern of even pixel.
7–0 PG_R_VALUE	Pattern generator R value. This field selects the R value for the generated pattern of even pixel.

42.2.3.7.6 CSI0 CCIR Code Register 1 (CSI0_CCIR_CODE_1)

Address 0xBASE+0xE030014 (CSI0_CCIR_CODE_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_CCIR_ERR_DET_EN	0	0	CSI0_STRT_FLD0_ACTV			CSI0_END_FLD0_ACTV		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CSI0_STRT_FLD0_BLNK_2ND			CSI0_END_FLD0_BLNK_2ND			CSI0_STRT_FLD0_BLNK_1ST			CSI0_END_FLD0_BLNK_1ST		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-164. CSI0 CCIR Code Register 1 (CSI0_CCIR_CODE_1)

Table 42-164. Register Field Descriptions

Field	Description
31–25	Reserved, should be cleared.
24 CSI0_CCIR_ERR_DET_EN	Enable error detection and correction for CCIR interlaced mode with protection bit. 0 Error detection and correction is disabled. 1 Error detection and correction is enabled.
23–22	Reserved, should be cleared.
21–19 CSI0_STRT_FLD0_ACTV	Start of field 0 active line command (interlaces mode). (In progressive mode, start of active line command mode).
18–16 CSI0_END_FLD0_ACTV	End of field 0 active line command (interlaces mode). (In progressive mode, end of active line command mode).
15–12	Reserved, should be cleared.
11–9 CSI0_STRT_FLD0_BLNK_2ND	Start of field 0 second blanking line command (interlaces mode). (In progressive mode this field is ignored).

Table 42-164. Register Field Descriptions (continued)

Field	Description
8–6 CSI0_END_FLD0_BLNK_2ND	End of field 0 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
5–3 CSI0_STRT_FLD0_BLNK_1ST	Start of field 0 first blanking line command (interlaces mode). (In progressive mode this field indicates start of blanking line command).
2–0 CSI0_END_FLD0_BLNK_1ST	End of field 0 first blanking line command (interlaces mode). (In progressive mode this field is ignored).

42.2.3.7.7 CSI0 CCIR Code Register 2 (CSI0_CCIR_CODE_2)

Address 0xBASE+0xE030018 (CSI0_CCIR_CODE_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	CSI0_STRT_FLD1_ACTV			CSI0_END_FLD1_ACTV		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CSI0_STRT_FLD1_BLNK_2ND			CSI0_END_FLD1_BLNK_2ND			CSI0_STRT_FLD1_BLNK_1ST			CSI0_END_FLD1_BLNK_1ST		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-165. CSI0 CCIR Code Register 2 (CSI0_CCIR_CODE_2)

Table 42-165. Register Field Descriptions

Field	Description
31–22	Reserved, should be cleared.
21–19 CSI0_STRT_FLD1_ACTV	Start of field 1 active line command (interlaces mode). (In progressive mode this field is ignored).
18–16 CSI0_STRT_FLD1_ACTV	End of field 1 active line command (interlaces mode). (In progressive mode this field is ignored).
15–12	Reserved, should be cleared.

Table 42-165. Register Field Descriptions (continued)

Field	Description
11–9 CSI0_START_FLD1_BLNK_2ND	Start of field 1 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
8–6 CSI0_END_FLD1_BLNK_2ND	End of field 1 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
5–3 CSI0_START_FLD1_BLNK_1ST	Start of field 1 first blanking line command (interlaces mode). (In progressive mode this field is ignored).
2–0 CSI0_END_FLD1_BLNK_1ST	End of field 1 first blanking line command (interlaces mode). (In progressive mode this field is ignored).

42.2.3.7.8 CSI0 CCIR Code Register 3 (CSI0_CCIR_CODE_3)

Address 0xBASE+0xE03001C (CSI0_CCIR_CODE_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CSI0_CCIR_PRECOM													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI0_CCIR_PRECOM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-166. CSI0 CCIR Code Register 3 (CSI0_CCIR_CODE_3)

Table 42-166. Register Field Descriptions

Field	Description
31–24	Reserved, should be cleared.
23–0 CSI0_CCIR_PRECOM	CCIR pre command. This field defines the sequence which comes before the CCIR command. For BT.656 the code should be written to bits [23:0] while bits [29:24] are ignored (3X8bit) For BT.1120 the code should be written to bits [29:0] (3X10bit)

42.2.3.7.9 CSI0 Data Identifier Register (CSI0_DI)

Address 0xBASE+0xE030020 (CSI0_DI) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI0_MIPI_DI3								CSI0_MIPI_DI2							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI0_MIPI_DI1								CSI0_MIPI_DI0							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 42-167. CSI0 Data Identifier Register (CSI0_DI)

Table 42-167. Register Field Descriptions

Field	Description
31–24 CSI0_MIPI_DI3	
23–16 CSI0_MIPI_DI2	
15–8 CSI0_MIPI_DI1	
7–0 CSI0_MIPI_DI0	

42.2.3.7.10 CSI0 SKIP Register (CSI0_SKIP)

This register controls the frame skipping supported between CSI0 and the SMFC.

Address 0xBASE+0xE030024 (CSI0_SKIP) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	CSI0_SKIP_ISP					CSI0_MAX_RATIO_SKIP_ISP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	CSI0_ID_2_SKIP		CSI0_SKIP_SMFC					CSI0_MAX_RATIO_SKIP_SMFC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-168. CSI0 SKIP Register (CSI0_SKIP)

Table 42-168. CSI0_SKIP Field Descriptions

Field	Description
31-24	Reserved
23–19 CSI0_SKIP_ISP	
18–16 CSI0_MAX_RATIO_SKIP_ISP	
15-10	Reserved
9-8 CSI0_ID_2_SKIP	Reserved

Table 42-168. CSI0_SKIP Field Descriptions (continued)

Field	Description
7–3 CSI0_SKIP_SMFC	CSI0 SKIP SMFC These 5 bits define the skipping pattern of the frames send to the SMFC. Skipping is done for a set of frames. The number of frames in a set is defined at CSI0_MAX_RATIO_SKIP_SMFC. when CSI0_MAX_RATIO_SKIP_SMFC = 1 => CSI0_SKIP_SMFC[0] is used; other bits are ignored when CSI0_MAX_RATIO_SKIP_SMFC = 2 => CSI0_SKIP_SMFC[1:0] are used; other bits are ignored when CSI0_MAX_RATIO_SKIP_SMFC = 3 => CSI0_SKIP_SMFC[2:0] are used; other bits are ignored when CSI0_MAX_RATIO_SKIP_SMFC = 4 => CSI0_SKIP_SMFC[3:0] are used; other bits are ignored when CSI0_MAX_RATIO_SKIP_SMFC = 5 => CSI0_SKIP_SMFC[4:0] are used; Setting bit #n of CSI0_SKIP_SMFC means that the #n frame in the set is skipped. For example: if CSI0_MAX_RATIO_SKIP_SMFC = 4 and CSI0_SKIP_SMFC = 11010 Frames #0 & Frame #2 will not be skipped as bit0 and bit2 are cleared Frames #1 & Frame #3 will be skipped as bit1 and bit3 are set bit #4 is ignored as CSI0_MAX_RATIO_SKIP_SMFC is set to 4
2–0 CSI0_MAX_RATIO_SKIP_SMFC	CSI0 Maximum Ratio Skip for SMFC These bits define the number of frames in a skipping set. The skipping number is equal to CSI0_MAX_RATIO_SKIP_SMFC+1; The maximum value of this bits is 5. When set to 0 the skipping is disabled.

42.2.3.7.11 CSI0 Compander Control Register (CSI0_CPD_CTRL)

Address 0xBASE+0xE030028 (CSI0_CPD_CTRL) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	CSI0_CPD			CSI0_RED_ROW_BEGIN	CSI0_GREEN_P_BEGIN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-169. CSI0 Compander Control Register (CSI0_CPD_CTRL)

Table 42-169. Register Field Descriptions

Field	Description
31–0	Reserved, should be cleared.
4–2 CSI0_CPD	CSI0_CPD These bits enable the compander in the path to different destination. CSI0_CPD[0] - CSI0_CPD[1] - Enable for the compander for data sent to the IC CSI0_CPD[2] - Enable for the compander for data sent to the IDMAC via SMFC If all the 3 bits are zero the compander is disabled
1 CSI0_RED_ROW_BEGIN	Color of first row in the frame. 0 First row in the frame is GBGB. 1 First row in the frame is GRGR.
0 CSI0_GREEN_P_BEGIN	Color of first component in the frame. 0 First component in the frame is blue or red, depending from RED_ROW bit. 1 First component in the frame is green

42.2.3.7.12 CSI0 Red component Compander Constants Register <i> (CSI0_CPD_RC_<i>)

These registers contain CONSTANT <i> parameters used for companding of red component.

Address 0xBASE+0xE03002C (CSI0_CPD_RC_<i>) Access: User read-write

offset 4

range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_CPD_RC_<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI0_CPD_RC_<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-170. CSI0 Red component Compander Constants Register <i> (CSI0_CPD_RC_<i>)

Table 42-170. CSI0_CPD_RC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONSTANT <2*i+1> parameter of Compander, Red component.
15-10	Reserved.
9-0	CONSTANT <2*i> parameter of Compander, Red component.

42.2.3.7.13 CSI0 Red component Compander SLOPE Register <i> (CSI0_CPD_RS_<i>)

These registers contain SLOPE <i> parameters used for companding of red component.

Address 0xBASE+0xE03004C (CSI0_CPD_RS_<i>) Access: User read-write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI0_CPD_RS<4*i+3>								CSI0_CPD_RS<4*i+2>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI0_CPD_RS<4*i+1>								CSI0_CPD_RS<4*i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-171. CSI0 Red component Compander SLOPE Register <i> (CSI0_CPD_RS_<i>)

Table 42-171. CSI0_CPD_RS_<i> Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, Red component.
23-16	SLOPE<4*i+2> parameter of Compander, Red component.
15-8	SLOPE<4*i+1> parameter of Compander, Red component.
7-0	SLOPE<4*i> parameter of Compander, Red component.

42.2.3.7.14 CSI0 GR component Compander Constants Register <i> (CSI0_CPD_GRC_<i>)

These registers contain CONSTANTi parameters used for companding of green components in GRGR rows.

Address 0xBASE+0xE03005C (CSI0_CPD_GRC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_CPD_GRC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI0_CPD_GRC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-172. CSI0 GR component Compander Constants Register <i> (CSI0_CPD_GRC_<i>)

Table 42-172. CSI0_CPD_GRC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONST<2*i+1> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRC should be equal to CSI0_CPD_GBC
15-10	Reserved.
9-0	CONSTANT<2*i> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRC should be equal to CSI0_CPD_GBC

42.2.3.7.15 CSI0 GR Component Compander SLOPE Register <i> (CSI0_CPD_GRS_<i>)

These registers contain SLOPEi parameters used for companding of green components in GRGR rows.

Address 0xBASE+0xE03007C (CSI0_CPD_GRS_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI0_CPD_GRS<4*i+3>								CSI0_CPD_GRS<4*i+2>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI0_CPD_GRS<4*i+1>								CSI0_CPD_GRS<4*i>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-173. CSI0 GR Component Compander SLOPE Register <i> (CSI0_CPD_GRS_<i>)

Table 42-173. CSI0_CPD_GRS_<i> Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRS should be equal to CSI0_CPD_GBS
23-16	SLOPE<4*i+2> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRS should be equal to CSI0_CPD_GBS
15-8	SLOPE<4*i+1> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRS should be equal to CSI0_CPD_GBS
7-0	SLOPE<4*i> parameter of Compander, GR component. If the input format is RGB/YUV then CSI0_CPD_GRS should be equal to CSI0_CPD_GBS

42.2.3.7.16 CSI0 GB component Compander Constants Register <i> (CSI0_CPD_GBC_<i>)

These registers contain CONSTANTi parameters used for companding of green components in GBGB rows.

Address 0xBASE+0xE03008C (CSI0_CPD_GBC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_CPD_GBC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI0_CPD_GBC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-174. CSI0 GB component Compander Constants Register <i> (CSI0_CPD_GBC_<i>)

Table 42-174. CSI0_CPD_GBC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONSTi+1 parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBC should be equal to CSI0_CPD_GRC

Table 42-174. CSI0_CPD_GBC_<i> Field Descriptions (continued)

Field	Description
15-10	Reserved.
9-0	CONSTANTi parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBC should be equal to CSI0_CPD_GRC

42.2.3.7.17 CSI0 GB component Compander SLOPE Register <i> (CSI0_CPD_GBS_<i>)

These registers contain SLOPEi parameters used for companding of green components in GBGB rows.

Address 0xBASE+0xE0300AC (CSI0_CPD_GBS_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI0_CPD_GBS<4*i+3>								CSI0_CPD_GBS<4*i+2>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI0_CPD_GBS<4*i+1>								CSI0_CPD_GBS<4*i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-175. CSI0 GB component Compander SLOPE Register <i> (CSI0_CPD_GBS_<i>)

Table 42-175. CSI0_CPD_GBS_<i> Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBS should be equal to CSI0_CPD_GRS
23-16	SLOPE<4*i+2> parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBS should be equal to CSI0_CPD_GRS
15-8	SLOPE<4*i+1> parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBS should be equal to CSI0_CPD_GRS
7-0	SLOPE<4*i> parameter of Compander, GB component. If the input format is RGB/YUV then CSI0_CPD_GBS should be equal to CSI0_CPD_GRS

42.2.3.7.18 CSI0 Blue component Compander Constants Register <i> (CSI0_CPD_BC_<i>)

These registers contain CONSTANTi parameters used for companding of blue component.

Address 0xBASE+0xE0300BC (CSI0_CPD_BC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI0_CPD_BC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI0_CPD_BC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-176. CSI0 Blue component Compander Constants Register <i> (CSI0_CPD_BC_<i>)

Table 42-176. CSI0_CPD_BC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONSTANT<2*i+1> parameter of Compander, Blue component.
15-10	Reserved.
10-0	CONSTANT<2*i> parameter of Compander, Blue component.

42.2.3.7.19 CSI0 Blue component Compander SLOPE Register <i> (CSI0_CPD_BS_<i>)

These registers contain SLOPEi parameters used for companding of red component.

Address 0xBASE+0xE0300DC (CSI0_CPD_BS_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI0_CPD_BS<4*i+3>								CSI0_CPD_BS<4*i+2>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI0_CPD_BS<4*i+1>								CSI0_CPD_BS<4*i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-177. CSI0 Blue component Compander SLOPE Register <i> (CSI0_CPD_BS_<i>)

Table 42-177. CSI0_CPD_BS_i Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, Blue component.
23-16	SLOPE<4*i+2> parameter of Compander, Blue component.
15-8	SLOPE<4*i+1> parameter of Compander, Blue component.
7-0	SLOPE<4*i> parameter of Compander, Blue component.

42.2.3.7.20 CSI0 Compander Offset Register 1 (CSI0_CPD_OFFSET1)

These registers contain Offset parameters used for companding.

Address 0xBASE+0xE0300EC (CSI0_CPD_OFFSET1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CSI0_CPD_B_OFFSET										CSI0_CPD_GB_OFFSET			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI0_CPD_GB_OFFSET								CSI0_CPD_GR_OFFSET							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-178. CSI0 Compander Offset Register 1 (CSI0_CPD_OFFSET1)

Table 42-178. CSI0_CPD_OFFSET1 Field Descriptions

Field	Description
31-30	Reserved
29-20 CSI0_CPD_B_OFFSET	CSI0 Blue component offset The value is between -512 to 511. The value is added to the blue component before companding. Clipping: If the result of the blue components value + the offset is smaller than 0, the result is zero If the result of the blue components value + the offset is greater than 1023, the result is 1023
19-10 CSI0_GB_OFFSET	CSI0 Green Blue component offset The value is between -512 to 511. The value is added to the blue component before companding. Clipping: If the result of the green-blue components value + the offset is smaller than 0, the result is zero If the result of the green-blue components value + the offset is greater than 1023, the result is 1023 If the input format is RGB/YUV then CSI1_GB_OFFSET must be equal to CSI1_GR_OFFSET
9-0 CSI0_GR_OFFSET	CSI0 Green Red component offset The value is between -512 to 511. The value is added to the green-red component before companding. Clipping: If the result of the green-red components value + the offset is smaller than 0, the result is zero If the result of the green-red components value + the offset is greater than 1023, the result is 1023

42.2.3.7.21 CSI0 Compander Offset Register 2 (CSI0_CPD_OFFSET2)

This register contain Offset parameters used for companding.

Address 0xBASE+0xE0300F0 (CSI0_CPD_OFFSET2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						CSI0_CPD_R_OFFSET									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-179. CSI0 Compander Offset Register 2 (CSI0_CPD_OFFSET2)

Table 42-179. CSI0_CPD_OFFSET2 Field Descriptions

Field	Description
31-10	Reserved
9-0 CSI0_CPD_R_OFFSET	CSI0 Red component offset The value is between -512 to 511. The value is added to the red component before companding. Clipping: If the result of the red components value + the offset is smaller than 0, the result is zero If the result of the red components value + the offset is greater than 1023, the result is 1023

42.2.3.8 CSI1 registers

42.2.3.8.1 CSI1 Sensor Configuration Register (CSI1_SENS_CONF)

Address 0xBASE+0xE038000 (CSI1_SENS_CONF) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI1_DATA_EN_POL	0	CSI1_FORCE_EOF	CSI1_JPEG_MODE	CSI1_JPEG8_EN	CSI1_DATA_DEST			CSI1_DIV_RATIO							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI1_EXT_VSYNC	CSI1_DATA_WIDTH				CSI1_SENS_DATA_FORMAT			CSI1_PACK_TIGHT	CSI1_SENS_PRTCL			CSI1_SENS_PIX_CLK_POL	CSI1_DATA_POL	CSI1_HSYNC_POL	CSI1_VSYNC_POL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-180. CSI1 Sensor Configuration Register (CSI1_SENS_CONF)

Table 42-180. Register Field Descriptions

Field	Description
31 CSI0_DATA_EN_POL	Invert IPP_IND_SENSB_DATA_EN input. This bit selects the polarity of IPP_IND_SENSB_DATA_EN signal. 0 IPP_IND_SENSB_DATA_EN is directly applied to internal circuitry. 1 IPP_IND_SENSB_DATA_EN is inverted before applied to internal circuitry.
30	Reserved, should be cleared.
29 CSI1_FORCE_EOF	Force End of frame This is a self clear bit allowing the user to force an End-of-frame event; This bit can be used in cases where the frame sent by the sensor was not completed. 1 - force end of frame 0 - no action
28 CSI1_JPEG_MODE	JPEG Mode - this bit defines the mode of the control signals when working in JPEG mode 1 - The data is valid as long as HSYNC and VSYNC signals are active; HSYNC is valid for single frame 0 - The frame starts withe the assertion of VSYNC. The frame ends on the next VSYNC or by setting the CSI0_FORCE_EOF bit
27 CSI1_JPEG8_EN	JPEG8 enable bit 1 - JPEG8 detection is enabled 0 - JPEG8 is disabled
26–24 CSI1_DATA_DEST	These bits enable the destination of the data coming from the CSI. CSI1_DATA_DEST[0] - CSI1_DATA_DEST[1] - destination is IC CSI1_DATA_DEST[2] - destination is IDMAC via SMFC
23–16 CSI1_DIV_RATIO	DIV Ratio Clock division ratio minus 1. This field defines the division ratio of HSP_CLK into SENSB_MCLK: SENSB_MCLK rate = HSP_CLK rate /(DIV_RATIO+1)
15 CSI1_EXT_VSYNC	External VSYNC enable. This bits select between external and internal VSYNC. 0 Internal VSYNC mode. 1 External VSYNC mode.
14–11 CSI1_DATA_WIDTH	Data width. This fields defines the number of bits per color. Values: 0000 - 4 bits per color 0001 - 8 bits per color 0010 - 9 bits per color 0011 - 10 bits per color 0100 - 11 bits per color 0101 - 12 bits per color 0110 - 13 bits per color 0111 - 14 bits per color 1000 - 15 bits per color 1001 - 16 bits per color

Table 42-180. Register Field Descriptions (continued)

Field	Description
10–8 CSI1_SENS_DATA_FORMAT	Data format from the sensor. This field defines the data format for the input of the CSI sensor. Values: 000 - full RGB or YUV444 001 - YUV422 (YUYV...) 010 - YUV422 (UYVY...) 011 - Bayer or Generic data 100 - RGB565 101 - RGB555 110 - RGB444 111 - JPEG
7 CSI1_PACK_TIGHT	CSI1 Pack Tight When the data format is YUV or RGB and the component's width is 9-16 bits, it can be sent to the memory in 2 different ways 1 - Three 10 bits components are packed into a 32 bit word. Color extension/reduction is performed 0 - Each component is written as a 16 bit word where the MSB is written to bit #15, color extension is done for the remaining least significant bits.
6–4 CSI1_SENS_PRTCL	Sensor protocol. This bit defines the sensor timing/data mode protocol. Values: 000 Gated clock mode 001 Non-gated clock mode 010 CCIR progressive mode (BT.656) 011 CCIR interlaced mode (BT.656) 100 CCIR progressive (BT.1120 DDR mode: data arrives on every edge of the clock) 101 CCIR progressive (BT.1120 SDR mode: data arrives only on the positive edge of the clock) 110 CCIR interlaced mode (BT.1120 DDR mode: data arrives on every edge of the clock) 111 CCIR interlaced mode (BT.1120 SDR mode: data arrives only on the positive edge of the clock)
3 CSI1_SENS_PIX_CLK_POL	Invert pixel clock input. This bit selects the polarity of pixel clock. 0 pixel clock is directly applied to internal circuitry. 1 pixel clock is inverted before applied to internal circuitry.
2 CSI1_DATA_POL	Invert data input. This bit selects the polarity of data input. 0 data lines are directly applied to internal circuitry. 1 data lines are inverted before applied to internal circuitry.
1 CSI1_HSYNC_POL	Invert IPP_IND_SENSB_HSYNC input. This bit selects the polarity of IPP_IND_SENSB_HSYNC signal. 0 IPP_IND_SENSB_HSYNC is directly applied to internal circuitry. 1 IPP_IND_SENSB_HSYNC is inverted before applied to internal circuitry.
0 CSI1_VSYNC_POL	Invert IPP_IND_SENSB_VSYNC input. This bit selects the polarity of IPP_IND_SENSB_VSYNC signal. 0 IPP_IND_SENSB_VSYNC is not inverted before applied to internal circuitry. 1 IPP_IND_SENSB_VSYNC is inverted before applied to internal circuitry.

42.2.3.8.2 CSI1 Sense Frame Size Register (CSI1_SENS_FRM_SIZE)

Address 0xBASE+0xE038004 (CSI1_SENS_FRM_SIZE) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	CSI1_SENS_FRM_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	CSI1_SENS_FRM_WIDTH												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-181. CSI1 Sense Frame Size Register (CSI1_SENS_FRM_SIZE)

Table 42-181. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 CSI1_SENS_FRM_HEIGHT	Sensor frame height minus 1. This field defines the sensor frame rows number minus 1.
15–13	Reserved, should be cleared.
12–0 CSI1_SENS_FRM_WIDTH	Sensor frame width minus 1. This field defines the sensor frame column number minus 1.

42.2.3.8.3 CSI1 Actual Frame Size Register (CSI1_ACT_FRM_SIZE)

Address 0xBASE+0xE038008 (CSI1_ACT_FRM_SIZE) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	CSI1_ACT_FRM_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	CSI1_ACT_FRM_WIDTH												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-182. CSI1 Actual Frame Size Register (CSI1_ACT_FRM_SIZE)

Table 42-182. Register Field Descriptions

Field	Description
31–28	Reserved, should be cleared.
27–16 CSI1_ACT_FRM_HEIGHT	Actual frame height minus 1. This field defines the CSI output frame rows number minus 1.
15–13	Reserved, should be cleared.
12–0 CSI1_ACT_FRM_WIDTH	Actual frame width minus 1. This field defines the CSI output frame columns number minus 1.

42.2.3.8.4 CSI1 Output Control Register (CSI1_OUT_FRM_CTRL)

Address 0xBASE+0xE03800C (CSI1_OUT_FRM_CTRL) Access: User read-write

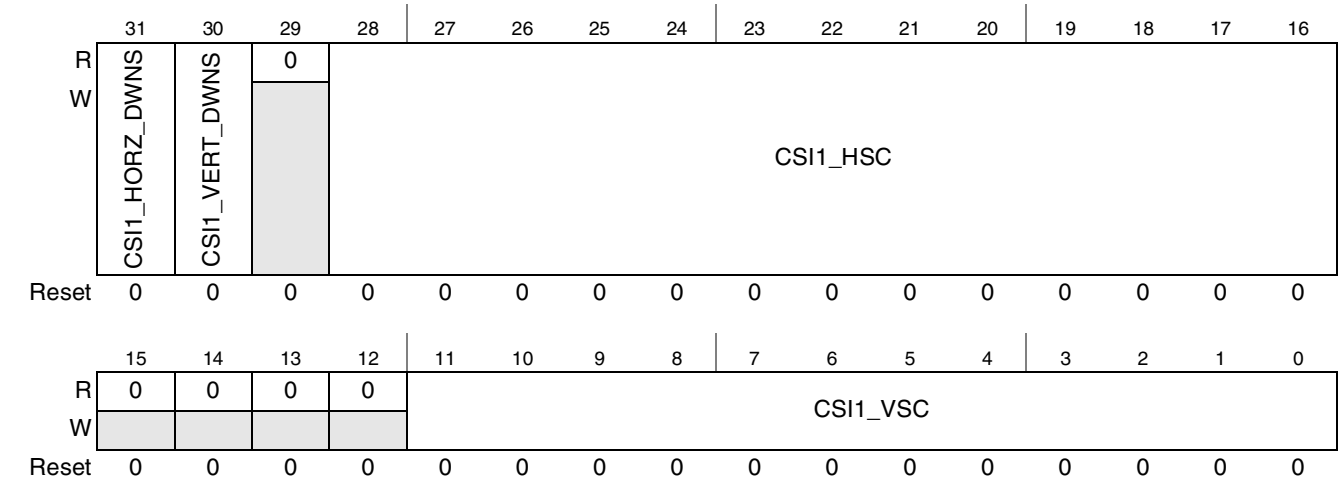


Figure 42-183. CSI1 Output Control Register (CSI1_OUT_FRM_CTRL)

Table 42-183. Register Field Descriptions

Field	Description
31 CSI1_HORZ_DWNS	Enable horizontal downsizing (decimation) by 2. 0 Downsizing disabled 1 Downsizing enabled
30 CSI1_VERT_DWNS	Enable vertical downsizing (decimation) by 2. 0 Downsizing disabled 1 Downsizing enabled

Table 42-183. Register Field Descriptions (continued)

Field	Description
28–16 CSI1_HSC	Horizontal skip. This field defines the number of columns to skip. In Interlaced mode this number refers to the number of lines per field
15-12	Reserved, should be cleared.
11–0 CSI1_VSC	Vertical skip. This field defines the number of rows to skip.

42.2.3.8.5 CSI1 Test Control Register (CSI1_TST_CTRL)

Address 0xBASE+0xE038010 (CSI1_TST_CTRL) Access: User read-write

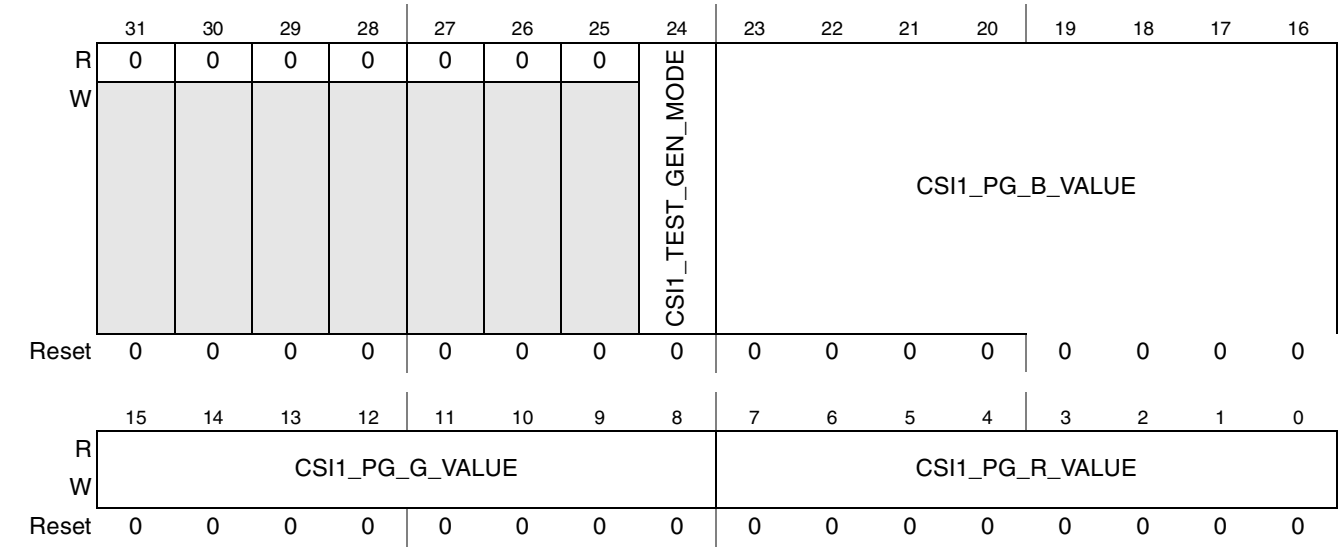


Figure 42-184. CSI1 Test Control Register (CSI1_TST_CTRL)

Table 42-184. Register Field Descriptions

Field	Description
31–25	Reserved, should be cleared.
2 TEST_GEN_MODE	Test generator mode. This bit activates the signal generation. 0 Test signal generator is inactive. 1 Test signal generator is active.
23–16 PG_B_VALUE	Pattern generator B value. This field selects the B value for the generated pattern of even pixel.

Table 42-184. Register Field Descriptions (continued)

Field	Description
15–8 PG_G_VALUE	Pattern generator G value. This field selects the G value for the generated pattern of even pixel.
7–0 PG_R_VALUE	Pattern generator R value. This field selects the R value for the generated pattern of even pixel.

42.2.3.8.6 CSI1 CCIR Code Register 1 (CSI1_CCIR_CODE_1)

Address 0xBASE+0xE038014 (CSI1_CCIR_CODE_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI1_CCIR_ERR_DET_EN	0	0	CSI1_STRT_FLD0_ACTV			CSI1_END_FLD0_ACTV		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CSI1_STRT_FLD0_BLNK_2ND			CSI1_END_FLD0_BLNK_2ND			CSI1_STRT_FLD0_BLNK_1ST			CSI1_END_FLD0_BLNK_1ST		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-185. CSI1 CCIR Code Register 1 (CSI1_CCIR_CODE_1)

Table 42-185. Register Field Descriptions

Field	Description
31–25	Reserved, should be cleared.
24 CSI1_CCIR_ERR_DET_EN	Enable error detection and correction for CCIR interlaced mode with protection bit. 0 Error detection and correction is disabled. 1 Error detection and correction is enabled.
23–22	Reserved, should be cleared.
21–19 CSI1_STRT_FLD0_ACTV	Start of field 0 active line command (interlaces mode). (In progressive mode, start of active line command mode).

Table 42-185. Register Field Descriptions (continued)

Field	Description
18–16 CSI1_END_FLD0_ACTV	End of field 0 active line command (interlaces mode). (In progressive mode, end of active line command mode).
15–12	Reserved, should be cleared.
11–9 CSI1_STRT_FLD0_BLNK_2ND	Start of field 0 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
8–6 CSI1_END_FLD0_BLNK_2ND	End of field 0 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
5–3 CSI1_STRT_FLD0_BLNK_1ST	Start of field 0 first blanking line command (interlaces mode). (In progressive mode this field indicates start of blanking line command).
2–0 CSI1_END_FLD0_BLNK_1ST	End of field 0 first blanking line command (interlaces mode). (In progressive mode this field is ignored).

42.2.3.8.7 CSI1 CCIR Code Register 2 (CSI1_CCIR_CODE_2)

Address 0xBASE+0xE038018 (CSI1_CCIR_CODE_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	CSI1_STRT_FLD1_ACTV			CSI1_END_FLD1_ACTV		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CSI1_STRT_FLD1_BLNK_2ND			CSI1_END_FLD1_BLNK_2ND			CSI1_STRT_FLD1_BLNK_1ST			CSI1_END_FLD1_BLNK_1ST		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-186. CSI1 CCIR Code Register 2 (CSI1_CCIR_CODE_2)

Table 42-186. Register Field Descriptions

Field	Description
31–22	Reserved, should be cleared.
21–19 CSI1_START_FLD1_ACTV	Start of field 1 active line command (interlaces mode). (In progressive mode this field is ignored).
18–16 CSI1_START_FLD1_ACTV	End of field 1 active line command (interlaces mode). (In progressive mode this field is ignored).
15–12	Reserved, should be cleared.
11–9 CSI1_START_FLD1_BLNK_2ND	Start of field 1 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
8–6 CSI1_END_FLD1_BLNK_2ND	End of field 1 second blanking line command (interlaces mode). (In progressive mode this field is ignored).
5–3 CSI1_START_FLD1_BLNK_1ST	Start of field 1 first blanking line command (interlaces mode). (In progressive mode this field is ignored).
2–0 CSI1_END_FLD1_BLNK_1ST	End of field 1 first blanking line command (interlaces mode). (In progressive mode this field is ignored).

42.2.3.8.8 CSI1 CCIR Code Register 3 (CSI1_CCIR_CODE_3)

Address 0xBASE+0xE03801C (CSI1_CCIR_CODE_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CSI1_CCIR_PRECOM													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI1_CCIR_PRECOM															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-187. CSI1 CCIR Code Register 3 (CSI1_CCIR_CODE_3)

Table 42-187. Register Field Descriptions

Field	Description
31–24	Reserved, should be cleared.
23–0 CSI1_CCIR_PRECOM	CCIR pre command. This field defines the sequence which comes before the CCIR command. For BT.656 the code should be written to bits [23:0] while bits [29:24] are ignored (3X8bit) For BT.1120 the code should be written to bits [29:0] (3X10bit)

42.2.3.8.9 CSI1 Data Identifier Register (CSI1_DI)

Address 0xBASE+0xE038020 (CSI1_DI) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI1_MIPI_DI3								CSI1_MIPI_DI2							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI1_MIPI_DI1								CSI1_MIPI_DI0							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 42-188. CSI1 Data Identifier Register (CSI1_DI)

Table 42-188. Register Field Descriptions

Field	Description
31–24 CSI1_MIPI_DI3	
23–16 CSI1_MIPI_DI2	
15–8 CSI0_MIPI_DI1	
7–0 CSI1_MIPI_DI0	

42.2.3.8.10 CSI1 SKIP Register (CSI1_SKIP)

This register control the frame skipping supported between CSI1 and the SMFC.

Address 0xBASE+0xE038024 (CSI1_SKIP) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	CSI1_SKIP_ISP					CSI1_MAX_RATIO_SKIP_ISP		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	CSI1_ID_2_SKIP		CSI1_SKIP_SMFC					CSI1_MAX_RATIO_SKIP_SMFC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-189. CSI1 SKIP Register (CSI1_SKIP)

Table 42-189. CSI1_SKIP Field Descriptions

Field	Description
31-24	Reserved
23–19 CSI1_SKIP_ISP	
18–16 CSI1_MAX_RATIO_SKIP_ISP	
17-10	Reserved
9-8 CSI1_ID_2_SKIP	Reserved

Table 42-189. CSI1_SKIP Field Descriptions (continued)

Field	Description
7–3 CSI1_SKIP_SMFC	CSI1 SKIP SMFC These 5 bits define the skipping pattern of the frames send to the SMFC. Skipping is done for a set of frames. The number of frames in a set is defined at CSI1_MAX_RATIO_SKIP_SMFC. when CSI1_MAX_RATIO_SKIP_SMFC = 1 => CSI1_SKIP_SMFC[0] is used; other bits are ignored when CSI1_MAX_RATIO_SKIP_SMFC = 2 => CSI1_SKIP_SMFC[1:0] are used; other bits are ignored when CSI1_MAX_RATIO_SKIP_SMFC = 3 => CSI1_SKIP_SMFC[2:0] are used; other bits are ignored when CSI1_MAX_RATIO_SKIP_SMFC = 4 => CSI1_SKIP_SMFC[3:0] are used; other bits are ignored when CSI1_MAX_RATIO_SKIP_SMFC = 5 => CSI1_SKIP_SMFC[4:0] are used; Setting bit #n of CSI1_SKIP_SMFC means that the #n frame in the set is skipped. For example: if CSI1_MAX_RATIO_SKIP_SMFC = 4 and CSI1_SKIP_SMFC = 11010 Frames #0 & Frame #2 will not be skipped as bit0 and bit2 are cleared Frames #1 & Frame #3 will be skipped as bit1 and bit3 are set bit #4 is ignored as CSI1_MAX_RATIO_SKIP_SMFC is set to 4
2–0 CSI1_MAX_RATIO_SKIP_SMFC	CSI1 Maximum Ratio Skip for SMFC These bits define the number of frames in a skipping set. These bits define the number of frames in a skipping set. The skipping number is equal to CSI1_MAX_RATIO_SKIP_SMFC+1; The maximum value of this bits is 5. When set to 0 the skipping is disabled.

42.2.3.8.11 CSI1 Compander Control Register (CSI1_CPD_CTRL)

Address 0xBASE+0xE038028 (CSI1_CPD_CTRL) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0			0	0	0	0	0	CSI1_CPD			CSI1_RED_ROW_BEGIN	CSI1_GREEN_P_BEGIN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-190. CSI1 Compander Control Register (CSI1_CPD_CTRL)

Table 42-190. Register Field Descriptions

Field	Description
31–0	Reserved, should be cleared.
4–2 CSI1_CPD	CSI0_CPD These bits enable the compander in the path to different destination. CSI1_CPD[0] - CSI1_CPD[1] - Enable for the compander for data sent to the IC CSI1_CPD[2] - Enable for the compander for data sent to the IDMAC via SMFC If all the 3 bits are zero the compander is disabled
1 CSI1_RED_ROW_BEGIN	Color of first row in the frame. 0 First row in the frame is GBGB. 1 First row in the frame is GRGR.
0 CSI1_GREEN_P_BEGIN	Color of first component in the frame. 0 First component in the frame is blue or red, depending from RED_ROW bit. 1 First component in the frame is green

42.2.3.8.12 CSI1 Red component Compander Constants Register <i> (CSI1_CPD_RC_<i>)

These registers contain CONSTANT <i> parameters used for companding of red component.

Address 0xBASE+0xE03802C (CSI1_CPD_RC_<i>) Access: User read-write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI1_CPD_RC_<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI1_CPD_RC_<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-191. CSI1 Red component Compander Constants Register <i> (CSI1_CPD_RC_<i>)

Table 42-191. CSI1_CPD_RC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONSTANT <2*i+1> parameter of Compander, Red component.
15-10	Reserved.
9-0	CONSTANT <2*i> parameter of Compander, Red component.

42.2.3.8.13 CSI1 Red component Compander SLOPE Register <i> (CSI1_CPD_RS_<i>)

These registers contain SLOPE <i> parameters used for companding of red component.

Address 0xBASE+0xE03804C (CSI1_CPD_RS_<i>) Access: User read-write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI1_CPD_RS<4*i+3>								CSI1_CPD_RS<4*i+2>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI1_CPD_RS<4*i+1>								CSI1_CPD_RS<4*i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-192. CSI1 Red component Compander SLOPE Register <i> (CSI1_CPD_RS_<i>)

Table 42-192. CSI1_CPD_RS_<i> Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, Red component.
23-16	SLOPE<4*i+2> parameter of Compander, Red component.
15-8	SLOPE<4*i+1> parameter of Compander, Red component.
7-0	SLOPE<4*i> parameter of Compander, Red component.

42.2.3.8.14 CSI1 GR component Compander Constants Register <i> (CSI1_CPD_GRC_<i>)

These registers contain CONSTANTi parameters used for companding of green components in GRGR rows.

Address 0xBASE+0xE03805C (CSI1_CPD_GRC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI1_CPD_GRC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI1_CPD_GRC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-193. CSI1 GR component Compander Constants Register <i> (CSI1_CPD_GRC_<i>)

Table 42-193. CSI1_CPD_GRC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16 CSI1_CPD_GRC<2*i+1>	CONST<2*i+1> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRC should be equal to CSI1_CPD_GBC
15-10	Reserved.
9-0 CSI1_CPD_GRC<2*i>	CONSTANT<2*i> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRC should be equal to CSI1_CPD_GBC

42.2.3.8.15 CSI1 GR Component Compander SLOPE Register <i> (CSI1_CPD_GRS_<i>)

These registers contain SLOPEi parameters used for companding of green components in GRGR rows.

Address 0xBASE+0xE03807C (CSI1_CPD_GRS_<i>) Access: User read/write
offset 4
range 0 .. 3

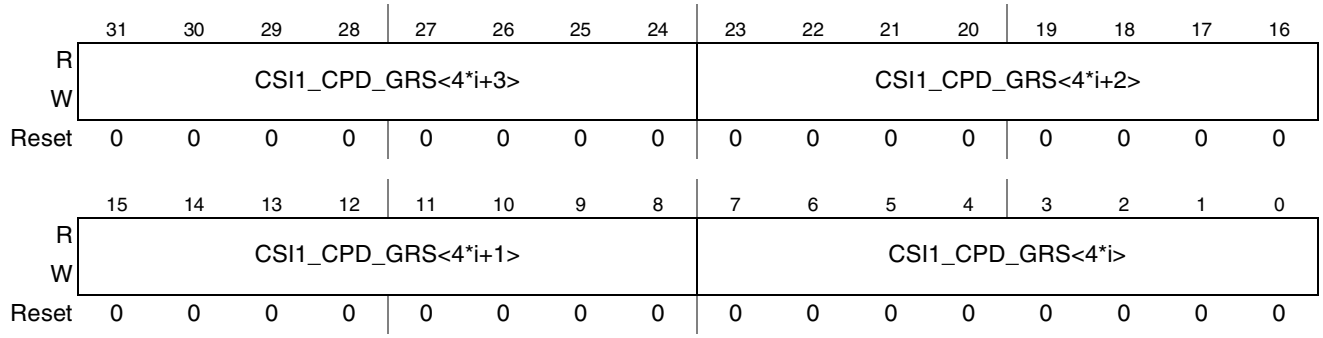


Figure 42-194. CSI1 GR Component Compander SLOPE Register <i> (CSI1_CPD_GRS_<i>)

Table 42-194. CSI1_CPD_GRS_<i> Field Descriptions

Field	Description
31-24 CSI1_CPD_GRS<4*i+3>	SLOPE<4*i+3> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRS should be equal to CSI1_CPD_GBS
23-16 CSI1_CPD_GRS<4*i+2>	SLOPE<4*i+2> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRS should be equal to CSI1_CPD_GBS
15-8 CSI1_CPD_GRS<4*i+1>	SLOPE<4*i+1> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRS should be equal to CSI1_CPD_GBS
7-0 CSI1_CPD_GRS<4*i>	SLOPE<4*i> parameter of Compander, GR component. If the input format is RGB/YUV then CSI1_CPD_GRS should be equal to CSI1_CPD_GBS

42.2.3.8.16 CSI1 GB component Compander Constants Register <i> (CSI1_CPD_GBC_<i>)

These registers contain CONSTANTi parameters used for companding of green components in GBGB rows.

Address 0xBASE+0xE03808C (CSI1_CPD_GBC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI1_CPD_GBC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI1_CPD_GBC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-195. CSI1 GB component Compander Constants Register <i> (CSI1_CPD_GBC_<i>)

Table 42-195. CSI1_CPD_GBC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16	CONSTi+1 parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBC should be equal to CSI1_CPD_GRC
15-10	Reserved.
9-0	CONSTANTi parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBC should be equal to CSI1_CPD_GRC

42.2.3.8.17 CSI1 GB component Compander SLOPE Register <i> (CSI1_CPD_GBS_<i>)

These registers contain SLOPEi parameters used for companding of green components in GBGB rows.

Address 0xBASE+0xE0380AC (CSI1_CPD_GBS_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSI1_CPD_GBS<4*i+3>								CSI1_CPD_GBS<4*i+2>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI1_CPD_GBS<4*i+1>								CSI1_CPD_GBS<4*i>							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-196. CSI1 GB component Compander SLOPE Register <i> (CSI1_CPD_GBS_<i>)

Table 42-196. CSI1_CPD_GBS_<i> Field Descriptions

Field	Description
31-24	SLOPE<4*i+3> parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBS should be equal to CSI1_CPD_GRS
23-16	SLOPE<4*i+2> parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBS should be equal to CSI1_CPD_GRS
15-8	SLOPE<4*i+1> parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBS should be equal to CSI1_CPD_GRS
7-0	SLOPE<4*i> parameter of Compander, GB component. If the input format is RGB/YUV then CSI1_CPD_GBS should be equal to CSI1_CPD_GRS

42.2.3.8.18 CSI1 Blue component Compander Constants Register <i> (CSI1_CPD_BC_<i>)

These registers contend CONSTANTi parameters used for companding of blue component.

Address 0xBASE+0xE0380BC (CSI1_CPD_BC_<i>) Access: User read/write
offset 4
range 0 .. 7

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	CSI1_CPD_BC<2*i+1>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CSI1_CPD_BC<2*i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-197. CSI1 Blue component Compander Constants Register <i> (CSI1_CPD_BC_<i>)

Table 42-197. CSI1_CPD_BC_<i> Field Descriptions

Field	Description
31-26	Reserved.
25-16 CSI1_CPD_BC<2*i+1>	CONSTANT<2*i+1> parameter of Compander, Blue component.

Table 42-197. CSI1_CPD_BC_<i> Field Descriptions (continued)

Field	Description
15-10	Reserved.
9-0 CSI1_CPD_BC<2*i>	CONSTANT<2*i> parameter of Compander, Blue component.

42.2.3.8.19 CSI1 Blue component Compander SLOPE Register <i> (CSI1_CPD_BS_<i>)

This registers contain SLOPEi parameters used for companding of red component.

Address 0xBASE+0xE0380DC (CSI1_CPD_BS_<i>) Access: User read/write
offset 4
range 0 .. 3

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CSI1_CPD_BS<4*i+3>								CSI1_CPD_BS<4*i+2>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CSI1_CPD_BS<4*i+1>								CSI1_CPD_BS<4*i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-198. CSI1 Blue component Compander SLOPE Register <i> (CSI1_CPD_BS_<i>)

Table 42-198. CSI1_CPD_BS_i Field Descriptions

Field	Description
31-24 CSI1_CPD_BS<4*i+3>	SLOPE<4*i+3> parameter of Compander, Blue component.
23-16 CSI1_CPD_BS<4*i+2>	SLOPE<4*i+2> parameter of Compander, Blue component.
15-8 CSI1_CPD_BS<4*i+1>	SLOPE<4*i+1> parameter of Compander, Blue component.
7-0 CSI1_CPD_BS<4*i>	SLOPE<4*i> parameter of Compander, Blue component.

42.2.3.8.20 CSI1 Compander Offset Register 1 (CSI1_CPD_OFFSET1)

These registers contain Offset parameters used for companding.

Address 0xBASE+0xE0380EC (CSI1_CPD_OFFSET1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	CSI1_CPD_B_OFFSET										CSI1_CPD_GB_OFFSET			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSI1_CPD_GB_OFFSET						CSI1_CPD_GR_OFFSET									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-199. CSI1 Compander Offset Register 1 (CSI1_CPD_OFFSET1)

Table 42-199. CSI1_CPD_OFFSET1 Field Descriptions

Field	Description
31-30	Reserved
29-20 CSI1_CPD_B_OFFSET	CSI1 Blue component offset The value is between -512 to 511. The value is added to the blue component before companding. Clipping: If the result of the blue components value + the offset is smaller than 0, the result is zero If the result of the blue components value + the offset is greater than 1023, the result is 1023
19-10 CSI1_CPD_GB_OFFSET	CSI1 Green Blue component offset The value is between -512 to 511. The value is added to the blue component before companding. Clipping: If the result of the green-blue components value + the offset is smaller than 0, the result is zero If the result of the green-blue components value + the offset is greater than 1023, the result is 1023 If the input format is RGB/YUV then CSI1_GB_OFFSET must be equal to CSI1_GR_OFFSET
9-0 CSI1_CPD_GR_OFFSET	CSI1 Green Red component offset The value is between -512 to 511. The value is added to the green-red component before companding. Clipping: If the result of the green-red components value + the offset is smaller than 0, the result is zero If the result of the green-red components value + the offset is greater than 1023, the result is 1023

42.2.3.8.21 CSI1 Compander Offset Register 2 (CSI1_CPD_OFFSET2)

These registers contain Offset parameters used for companding.

Address 0xBASE+0xE0380F0 (CSI1_CPD_OFFSET2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						CSI1_CPD_R_OFFSET									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-200. CSI1 Compander Offset Register 2 (CSI1_CPD_OFFSET2)

Table 42-200. CSI1_CPD_OFFSET2 Field Descriptions

Field	Description
31-10	Reserved
9-0 CSI1_CPD_R_OFFSET	CSI1 Red component offset The value is between -512 to 511. The value is added to the red component before companding. Clipping: If the result of the red components value + the offset is smaller than 0, the result is zero If the result of the red components value + the offset is greater than 1023, the result is 1023

42.2.3.9 DI0 Registers

42.2.3.9.1 DI0 General Register (DI0_GENERAL)

Address 0xBASE+0xE040000 (DI0_GENERAL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	di0_pin8_pin15_sel	di0_disp_y_sel			DI0_CLOCK_STOP_MODE				DI0_DISP_CLOCK_INIT	di0_mask_sel	di0_vsync_ext	di0_clk_ext	DI0_WATCHDOG_MODE		di0_polarity_disp_clk	0
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	di0_sync_count_sel				di0_err_treatment	di0_erm_vsync_sel	di0_polarity_cs1	di0_polarity_cs0	di0_polarity_8	di0_polarity_7	di0_polarity_6	di0_polarity_5	di0_polarity_4	di0_polarity_3	di0_polarity_2	di0_polarity_1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-201. DI0 General Register (DI0_GENERAL)

Table 42-201. DI0_GENERAL Field Descriptions

Field	Description
31 di0_pin8_pin15_sel	This bit routes PIN8 over PIN15 1 - PIN8 is routed to PIN15, PIN8 is also routed to PIN8 0 - PIN15 is routed to PIN15, PIN8 is routed to PIN8
30–28 di0_disp_y_sel	DI0 Display Vertical coordinate (Y) select. This field defines which one of the 8 counters will be used as a display's line counter. 000 counter #1 is selected ... 111 counter #8 is selected
27–24 DI0_CLOCK_STOP_MODE	DI clock stop mode When performing a clock change. The DI stops the clock to the display. These field defines when the clock will be stopped 0000 - stop at the next edge of the display clock 0001-1001 stop at the next event of one of the counters (counter #1 to counter #9) 1100 - stop at EOL (end of a line), but if stop request is during blanking interval, stop now 1101 - stop at EOF (end of a frame), but if stop request is during blanking interval, stop now 1110 - stop at EOL (end of a line), but if stop request is during blanking interval, stop at the end of the next line 1111 - stop at EOF (end of a frame), but if stop request is during blanking interval, stop at the end of the next frame Stopping at EOL/EOF is supported for the case where the data is coming from the IDMAC (DMA access). In case that only direct accesses is performed, the user should set this field to 0000

Table 42-201. DI0_GENERAL Field Descriptions (continued)

Field	Description
23 DI0_DISP_CLOCK_INIT	Display clock's initial mode For synchronization error conditions the display clock can be stopped on the next VSYNC 1 - The display's clock is running after the next VSYNC (indicating new frame) 0 - The display's clock is stopped after the next VSYNC (indicating new frame)
22 di0_mask_sel	DI0 Mask select. IPP_PIN_2 output of the DI that functions as MASK signal can come from 2 sources: counter #2 or extracted from the MASK data coming from the memory. 1 IPP_PIN_2 is coming from extracted MASK data coming from the memory 0 IPP_PIN_2 is coming from counter #2
21 di0_vsync_ext	DI0 External VSYNC. This bit selects the source of the VSYNC signal 1 External to the IPUv3EX 0 Internally generated by the IPUv3EX
20 di0_clk_ext	DI0 External Clock. This bit selects the source of the DI0's clock 1 The source of the clock is external to the IPUv3EX 0 The clock is internally generated by the IPUv3EX
19-18 DI0_WATCHDOG_MODE	DI0 watchdog mode In case of a display error where the DI clock is stopped (defined at di0_err_treatment). An internal watchdog counts DI clocks. If this timer reached its pre defined value the DI will skip the current frame and restart on the frame. This 2 bits define the number of DI clock cycles that the timer counts. 00 The timer counts 4 DI cycles 01 The timer counts 16 DI cycles 10 The timer counts 64 DI cycles 11 The timer counts 128 DI cycles
17 di0_polarity_disp_clk	DI0 Output Clock's polarity This bits define the polarity of the DI0's clock. 1 The output clock is active high 0 The output clock is active low
16	Reserved
15–12 di0_sync_count_sel	For synchronous flow error: selects synchronous flow synchronization counter in DI:
11 di0_err_treatment	In case of synchronous flow error there are 2 ways to handle the display 1 to wait (i.e. stop clock) 0 Drive the last component
10 di0_erm_vsync_sel	DI0 error recovery module's VSYNC source select The error recovery module detect a case where the DI's VSYNC is asserted before the EOF. This bit selects the source of the VSYNC signal monitored by this mechanism. 1- vsync_post - an internal VSYNC signal asserted 2 lines after the DI's VSYNC 0 - vsync_pre - an internal VSYNC signal asserted 2 lines before the DI's VSYNC

Table 42-201. DI0_GENERAL Field Descriptions (continued)

Field	Description
9 di0_polarity_cs1	DI0 Chip Select's 1 polarity This bits define the polarity of the DI's CS1. 1 The CS1 is active high 0 The CS1 is active low
8 di0_polarity_cs0	DI0 Chip Select's 0 polarity This bits define the polarity of the DI's CS0. 1 The CS0 is active high 0 The CS0 is active low
7-0 di0_polarity_<i+1>	DI0 output pin's polarity This bits define the polarity of each of the DI's outputs. 1 The output pin is active high 0 The output pin is active low

42.2.3.9.2 DI0 Base Sync Clock Gen 0 Register (DI0_BS_CLKGEN0)

Address 0xBASE+0xE040004 (DI0_BS_CLKGEN0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di0_disp_clk_offset								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	di0_disp_clk_period											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-202. DI0 Base Sync Clock Gen 0 Register (DI0_BS_CLKGEN0)

Table 42-202. DI0_BS_CLKGEN0 Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_disp_clk_offset	DI0 Display Clock Offset The DI has the ability to delay the display's clock This field defines the amount of IPUv3EX's clock cycles added as delay on this clock.
15–12	Reserved.
11–0 di0_disp_clk_period	DI0 Display Clock Period This field defines the Display interface clock period for display write access. This parameter contains an integer part (bits 11:4) and a fractional part (bits 3:0). It defines a fractional division ratio of the HSP_CLK clock for generation of the display's interface clock.

42.2.3.9.3 DI0 Base Sync Clock Gen 1 Register (DI0_BS_CLKGEN1)

Address 0xBASE+0xE040008 (DI0_BS_CLKGEN1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di0_disp_clk_down								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	di0_disp_clk_up								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-203. DI0 Base Sync Clock Gen 1 Register (DI0_BS_CLKGEN1)

Table 42-203. DI0_BS_CLKGEN1 Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_disp_clk_down	DI0 display clock falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is a time interval between display's access start point and display 's interface clock falling edge.
15–9	Reserved.
8–0 di0_disp_clk_up	DI0 display clock rising edge position This parameter contains an integer part (bits 8:1) and a fractional part (bit 0). The position value is a time interval between display's access start point and display 's interface clock rising edge.

42.2.3.9.4 DI0 Sync Wave Gen 1 Register 0 (DI0_SW_GEN0_1)

Address 0xBASE+0xE04000C (DI0_SW_GEN0_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_1												di0_run_resolution_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_1												di0_offset_resolution_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-204. DI0 Sync Wave Gen 1 Register 0 (DI0_SW_GEN0_1)

Table 42-204. DI0_SW_GEN0_1 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_1	DI0 counter #1 pre defined value This fields defines the counter #1 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_1 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_1	DI0 counter #1 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - NA 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di0_offset_value_1	DI0 counter #1 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_1	DI0 counter #1 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock 010 - NA 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.9.5 DI0 Sync Wave Gen 2 Register 0 (DI0_SW_GEN0_2)

Address 0xBASE+0xE040010 (DI0_SW_GEN0_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_2												di0_run_resolution_2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_2												di0_offset_resolution_2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-205. DI0 Sync Wave Gen 2 Register 0 (DI0_SW_GEN0_2)

Table 42-205. DI0_SW_GEN0_2 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_2	DI0 counter #2 pre defined value This fields defines the counter #2 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_2 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_2	DI0 counter #2 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock 010 - The Counter is triggered by counter #1 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di0_offset_value_2	DI0 counter #2 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_2	DI0 counter #2 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit.110 - External VSYNC 111 Counter is always on.

42.2.3.9.6 DI0 Sync Wave Gen 3 Register 0 (DI0_SW_GEN0_3)

Address 0xBASE+0xE040014 (DI0_SW_GEN0_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_3												di0_run_resolution_3		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_3												di0_offset_resolution_3		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-206. DI0 Sync Wave Gen 3 Register 0 (DI0_SW_GEN0_3)

Table 42-206. DI0_SW_GEN0_3 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_3	DI0 counter #3 pre defined value This fields defines the counter #3 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_3 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_3	DI0 counter #3 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di0_offset_value_3	DI0 counter #3 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_3	DI0 counter #3 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.9.7 DI0 Sync Wave Gen 4 Register 0 (DI0_SW_GEN0_4)

Address 0xBASE+0xE040018 (DI0_SW_GEN0_4) Access: User read/write

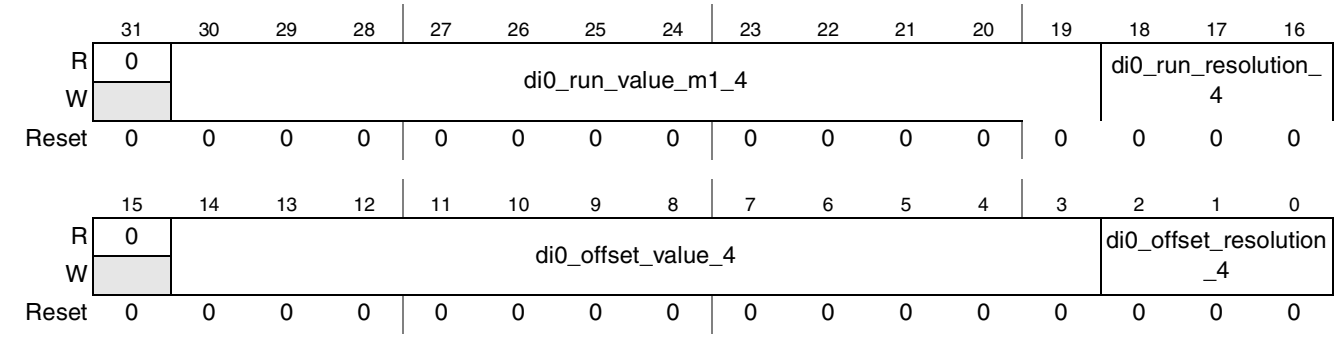


Figure 42-207. DI0 Sync Wave Gen 4 Register 0 (DI0_SW_GEN0_4)

Table 42-207. DI0_SW_GEN0_4 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_4	DI0 counter #4 pre defined value This fields defines the counter #4 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_4 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_4	DI0 counter #4 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

Table 42-207. DI0_SW_GEN0_4 Field Descriptions (continued)

Field	Description
14–3 di0_offset_value_4	DI0 counter #4 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_4	DI0 counter #4 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.9.8 DI0 Sync Wave Gen 5 Register 0 (DI0_SW_GEN0_5)

Address 0xBASE+0xE04001C (DI0_SW_GEN0_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_5												di0_run_resolution_5		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_5												di0_offset_resolution_5		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-208. DI0 Sync Wave Gen 5 Register 0 (DI0_SW_GEN0_5)

Table 42-208. DI0_SW_GEN0_5 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_5	DI0 counter #5 pre defined value This fields defines the counter #5 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_5 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.

Table 42-208. DI0_SW_GEN0_5 Field Descriptions (continued)

Field	Description
18–16 di0_run_resolution_5	DI0 counter #5 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.
14–3 di0_offset_value_5	DI0 counter #5 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_5	DI0 counter #5 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 -The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.

42.2.3.9.9 DI0 Sync Wave Gen 6 Register 0 (DI0_SW_GEN0_6)

Address 0xBASE+0xE040020 (DI0_SW_GEN0_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_6												di0_run_resolution_6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_6												di0_offset_resolution_6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-209. DI0 Sync Wave Gen 6 Register 0 (DI0_SW_GEN0_6)

Table 42-209. DI0_SW_GEN0_6 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_6	DI0 counter #6 pre defined value This fields defines the counter #6 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_6 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_6	DI0 counter #6 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di0_offset_value_6	DI0 counter #6 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_6	DI0 counter #6 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.9.10 DI0 Sync Wave Gen 7 Register 0 (DI0_SW_GEN0_7)

Address 0xBASE+0xE040024 (DI0_SW_GEN0_7) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_7												di0_run_resolution_7		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_7												di0_offset_resolution_7		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-210. DI0 Sync Wave Gen 7 Register 0 (DI0_SW_GEN0_7)

Table 42-210. DI0_SW_GEN0_7 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_7	DI0 counter #7 pre defined value This fields defines the counter #7 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_7 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_7	DI0 counter #1 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di0_offset_value_7	DI0 counter #7 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_1	DI0 counter #7 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.9.11 DI0 Sync Wave Gen 8 Register 0 (DI0_SW_GEN0_8)

Address 0xBASE+0xE040028 (DI0_SW_GEN0_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_8												di0_run_resolution_8		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_8												di0_offset_resolution_8		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-211. DI0 Sync Wave Gen 8 Register 0 (DI0_SW_GEN0_8)

Table 42-211. DI0_SW_GEN0_8 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_8	DI0 counter #8 pre defined value This fields defines the counter #8 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_8 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_8	DI0 counter #8 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di0_offset_value_8	DI0 counter #8 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_8	DI0 counter #8 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.9.12 DI0 Sync Wave Gen 9 Register 0 (DI0_SW_GEN0_9)

Address 0xBASE+0xE04002C (DI0_SW_GEN0_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_run_value_m1_9												di0_run_resolution_9		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_offset_value_9												di0_offset_resolution_9		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-212. DI0 Sync Wave Gen 9 Register 0 (DI0_SW_GEN0_9)

Table 42-212. DI0_SW_GEN0_9 Field Descriptions

Field	Description
31	Reserved.
30–19 di0_run_value_m1_9	DI0 counter #9 pre defined value This fields defines the counter #9 pre defines value. When the counter is auto reload mode (di0_cnt_auto_reload_9 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di0_step_repeat field.
18–16 di0_run_resolution_9	DI0 counter #9 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di0_offset_value_9	DI0 counter #9 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di0_offset_resolution_9	DI0 counter #9 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.9.13 DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Address 0xBASE+0xE040030 (DI0_SW_GEN1_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_1		di0_cnt_auto_reload_1	di0_cnt_clr_sel_1			di0_cnt_down_1								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_1			di0_cnt_polarity_clr_sel_1			di0_cnt_up_1								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-213. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-213. DI0_SW_GEN1_1 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_1	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_1	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_1 field
27–25 di0_cnt_clr_sel_1	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di0_cnt_down_1	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.

Table 42-213. DI0_SW_GEN1_1 Field Descriptions (continued)

Field	Description
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_1	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–9 di0_cnt_polarity_clr_sel_1	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Reserved 011 - Reserved 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di0_cnt_up_1	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.14 DI0 Sync Wave 2 Gen Register 1 (DI0_SW_GEN1_2)

Address 0xBASE+0xE040034 (DI0_SW_GEN1_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_2			di0_cnt_auto_reload_2	di0_cnt_clr_sel_2			di0_cnt_down_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_2			di0_cnt_polarity_clr_sel_2				di0_cnt_up_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-214. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-214. DI0_SW_GEN1_2 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_2	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_2	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_2	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di0_cnt_down_2	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_2	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

Table 42-214. DI0_SW_GEN1_2 Field Descriptions (continued)

Field	Description
11–9 di0_cnt_polarity_clr_sel_2	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Reserved 011 - Reserved 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di0_cnt_up_2	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.15 DI0 Sync Wave 3 Gen Register 1 (DI0_SW_GEN1_3)

Address 0xBASE+0xE040038 (DI0_SW_GEN1_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_3			di0_cnt_auto_reload_3	di0_cnt_clr_sel_3			di0_cnt_down_3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_3			di0_cnt_polarity_clr_sel_3				di0_cnt_up_3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-215. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-215. DI0_SW_GEN1_3 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_3	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_3	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field

Table 42-215. DI0_SW_GEN1_3 Field Descriptions (continued)

Field	Description
27–25 di0_cnt_clr_sel_3	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di0_cnt_down_3	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_3	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–9 di0_cnt_polarity_clr_sel_3	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di0_cnt_up_3	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.16 DI0 Sync Wave 4 Gen Register 1 (DI0_SW_GEN1_4)

Address 0xBASE+0xE04003C (DI0_SW_GEN1_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_4		di0_cnt_auto_reload_4	di0_cnt_clr_sel_4			di0_cnt_down_4								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_4			di0_cnt_polarity_clr_sel_4			di0_cnt_up_4								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-216. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-216. DI0_SW_GEN1_4 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_4	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_4	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_4	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 -CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di0_cnt_down_4	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved

Table 42-216. DI0_SW_GEN1_4 Field Descriptions (continued)

Field	Description
14–12 di0_cnt_polarity_trigger_sel_4	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–9 di0_cnt_polarity_clr_sel_4	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di0_cnt_up_4	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.17 DI0 Sync Wave 5 Gen Register 1 (DI0_SW_GEN1_5)

Address 0xBASE+0xE040040 (DI0_SW_GEN1_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_5		di0_cnt_auto_reload_5	di0_cnt_clr_sel_5				di0_cnt_down_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_5			di0_cnt_polarity_clr_sel_5			di0_cnt_up_5								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-217. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-217. DI0_SW_GEN1_5 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_5	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_5	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_5	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.
24–16 di0_cnt_down_5	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_5	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.

Table 42-217. DI0_SW_GEN1_5 Field Descriptions (continued)

Field	Description
11–9 di0_cnt_polarity_clr_sel_5	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Reserved 111 - Reserved
8–0 di0_cnt_up_5	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.18 DI0 Sync Wave 6 Gen Register 1 (DI0_SW_GEN1_6)

Address 0xBASE+0xE040044 (DI0_SW_GEN1_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_6		di0_cnt_auto_reload_6	di0_cnt_clr_sel_6				di0_cnt_down_6							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_6			di0_cnt_polarity_clr_sel_6				di0_cnt_up_6							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-218. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-218. DI0_SW_GEN1_6 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_6	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_6	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field

Table 42-218. DI0_SW_GEN1_6 Field Descriptions (continued)

Field	Description
27–25 di0_cnt_clr_sel_6	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di0_cnt_down_6	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_6	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
11–9 di0_cnt_polarity_clr_sel_6	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Reserved
8–0 di0_cnt_up_6	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.19 DI0 Sync Wave 7 Gen Register 1 (DI0_SW_GEN1_7)

Address 0xBASE+0xE040048 (DI0_SW_GEN1_7) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_7		di0_cnt_auto_reload_7	di0_cnt_clr_sel_7			di0_cnt_down_7								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_7			di0_cnt_polarity_clr_sel_7			di0_cnt_up_7								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-219. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-219. DI0_SW_GEN1_7 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_7	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_7	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_7	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di0_cnt_down_7	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved

Table 42-219. DI0_SW_GEN1_7 Field Descriptions (continued)

Field	Description
14–12 di0_cnt_polarity_trigger_sel_7	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
11–9 di0_cnt_polarity_clr_sel_7	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Output is inverted if the output of counter #6 is set
8–0 di0_cnt_up_7	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.20 DI0 Sync Wave 8 Gen Register 1 (DI0_SW_GEN1_8)

Address 0xBASE+0xE04004C (DI0_SW_GEN1_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di0_cnt_polarity_gen_en_8		di0_cnt_auto_reload_8	di0_cnt_clr_sel_8				di0_cnt_down_8							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di0_cnt_polarity_trigger_sel_8			di0_cnt_polarity_clr_sel_8			di0_cnt_up_8								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-220. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-220. DI0_SW_GEN1_8 Field Descriptions

Field	Description
31	Reserved.
30–29 di0_cnt_polarity_gen_en_8	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_8	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_8	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di0_cnt_down_8	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di0_cnt_polarity_trigger_sel_8	DI0 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

Table 42-220. DI0_SW_GEN1_8 Field Descriptions (continued)

Field	Description
11–9 di0_cnt_polarity_clr_sel_8	DI0 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Output is inverted if the output of counter #6 is set
8–0 di0_cnt_up_8	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.21 DI0 Sync Wave 9 Gen Register 1 (DI0_SW_GEN1_9)

Address 0xBASE+0xE040050 (DI0_SW_GEN1_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	di0_gentime_sel_9			di0_cnt_auto_reload_9	di0_cnt_clr_sel_9				di0_cnt_down_9							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	di0_tag_sel_9	0	0	0	0	0	0	di0_cnt_up_9								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-221. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-221. DI0_SW_GEN1_9 Field Descriptions

Field	Description
31-29 di0_gentime_sel_9	Counter #9 main waveform select This field defines the counter that counter #9's auxiliary waveform will be attached too. 000 - Counter #9's waveform is attached to counter #1's waveform 001 - Counter #9's waveform is attached to counter #2's waveform 010 - Counter #9's waveform is attached to counter #3's waveform 011 - Counter #9's waveform is attached to counter #4's waveform 100 - Counter #9's waveform is attached to counter #5's waveform 101 - Counter #9's waveform is attached to counter #6's waveform 110 - Counter #9's waveform is attached to counter #7's waveform 111 - Counter #9's waveform is attached to counter #8's waveform
30–29 di0_cnt_polarity_gen_en_9	DI0 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di0_cnt_auto_reload_9	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di0_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di0_step_repeat_<i> field
27–25 di0_cnt_clr_sel_9	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di0_cnt_down_9	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15 di0_tag_sel_9	Counter #9 can send a synchronous tag when counter #9 reach its predefined value or when it's triggering counter reaches its pre defined value. 1 - tag source is counter #9 0 - Tag's source is the triggering counter.
14-9	Reserved
8–0 di0_cnt_up_9	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.9.22 DI0 Sync Assistance Gen Register (DI0_SYNC_AS_GEN)

Address 0xBASE+0xE040054 (DI0_SYNC_AS_GEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	di0_sync_start_en	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	di0_vsync_sel			0	di0_sync_start											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-222. DI0 Sync Assistance Gen Register (DI0_SYNC_AS_GEN)

Table 42-222. DI0_SYNC_AS_GEN Field Descriptions

Field	Description
31–29	Reserve
28 di0_sync_start_en	Reserve
27–16	Reserve
15–13 di0_vsync_sel	VSYNC select This field defines which of the counters functions as VSYNC signal 000 - VSYNC is coming from counter #1 001 - VSYNC is coming from counter #2 ... 111 - VSYNC is coming from counter #8
12	Reserved.
11–0 di0_sync_start	DI0 Sync start This field defines the number of low (including blanking rows) on the which the DI0 starts preparing the data for the next frame.

42.2.3.9.23 DI0 Data Wave Gen <i> Registers (DI0_DW_GEN_<i>)

The DI0_DW_GEN_<i> register holds pointers for the waveform generators.

These registers have different bit arrangements for parallel and serial display. When using a parallel display Table 42-223 is applicable. When using a serial interface Table 42-224 is applicable

Address 0xBASE+0xE040058 (DI0_DW_GEN_<i>) Access: User read/write
offset 4
range 0 .. 11

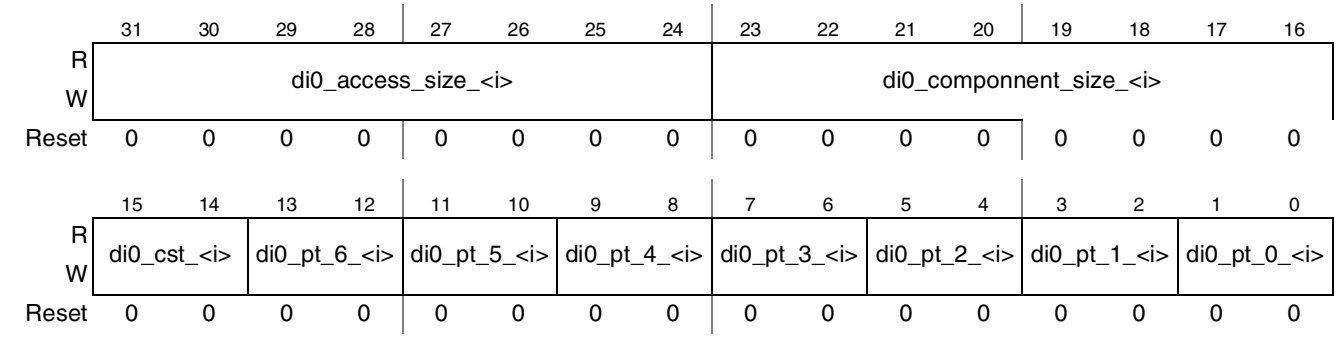


Figure 42-223. DI0 Data Wave Gen <i> Registers (DI0_DW_GEN_<i>)

Table 42-223. DI0_DW_GEN <i> Field Descriptions for parallel display

Field	Description
31–24 di0_access_size_<i>	DI0 Access Size <i> This field defines the amount of IPUv3EX cycles between any 2 accesses (an access may be a pixel or generic data that may have more one component)
23–16 di0_componnent_size_<i>	DI0 component Size This field defines the amount of IPUv3EX cycles between any 2 components
15-14 di0_cst_<i>	DI0 Chip Select pointer for waveform <i> This field points to a register that defines the waveform of the CS pin. The CS is automatically mapped to a specific display 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
13-12 di0_pt_6_<i>	DI0 PIN_17 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_17 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
11-10 di0_pt_5_<i>	DI0 PIN_16 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_16 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>

Table 42-223. DI0_DW_GEN <i> Field Descriptions for parallel display

Field	Description
9-8 di0_pt_4 _<i>	DI0 PIN_15 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_15 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
7-6 di0_pt_3 _<i>	DI0 PIN_14 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_14 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
5-4 di0_pt_2 _<i>	DI0 PIN_13 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_13 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
3-2 di0_pt_1 _<i>	DI0 PIN_12 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_12 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
1-0 di0_pt_0 _<i>	DI0 PIN_11 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_11 pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>

Table 42-224. DI0_DW_GEN <i> Field Descriptions for serial display

Field	Description
31–24 di0_seria l_period_ <i>	DI0 Serial Period <i> This field defines the period of the time base serial display clock. The units are the internal DI clock
23–16 di0_start _period_ <i>	DI0 start period This field defines the amount of cycles between the point where the access is ready to be launched to the actual point where the time base serial display clock restarts. The units are the internal DI clock

Table 42-224. DI0_DW_GEN <i> Field Descriptions for serial display

Field	Description
15-14 di0_cst_<i>	DI0 Chip Select pointer for waveform <i> This field points to a register that defines the waveform of the CS pin. For serial displays the down value as defined on DI0_DW_SET*_<i> is measured from the assertion of the last serial display time base clock. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
13-9	Reserved
8-4 di0_serial_valid_bits<i>	DI0 Serial valid bits. This field defines the amount of valid bits to be transmitted within the 32 bits internal word aligned to bit[0]. The actual amount of valid bits is di0_serial_valid_bits_<i> + 1
3-2 di0_serial_rs_<i>	DI0 Serial RS This field points to a register that defines the waveform of the RS pin. For serial displays the down value as defined on DI0_DW_SET*_<i> is measured from the assertion of the last serial display time base clock. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>
1-0 di0_serial_clk_<i>	DI0 serial clock<i> This field points to a register that defines the waveform of the Serial clock pin. 00 - The waveform is defined according to the settings on DI0_DW_SET0_<i> 01- The waveform is defined according to the settings on DI0_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI0_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI0_DW_SET3_<i>

42.2.3.9.24 DI0 Data Wave Set 0 <i> Registers (DI0_DW_SET0_<i>)

Address 0xBASE+0xE040088 (DI0_DW_SET0_<i>) Access: User read/write
offset 4
range 0 .. 11

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di0_data_cnt_down0_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	di0_data_cnt_up0_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-224. DI0 Data Wave Set 0 <i> Registers (DI0_DW_SET0_<i>)

Table 42-225. DI0_DW_SET0_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_data_cnt_down0_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a point according to the corresponding di0_pt_*_<i>
15–9	Reserved.
8–0 di0_data_cnt_up0_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a point according to the corresponding di0_pt_*_<i>

42.2.3.9.25 DI0 Data Wave Set 1 <i> Registers (DI0_DW_SET1_<i>)

Address 0xBASE+0xE0400B8 (DI0_DW_SET1_<i>)
offset 4
range 0 .. 11

Access: User read/write

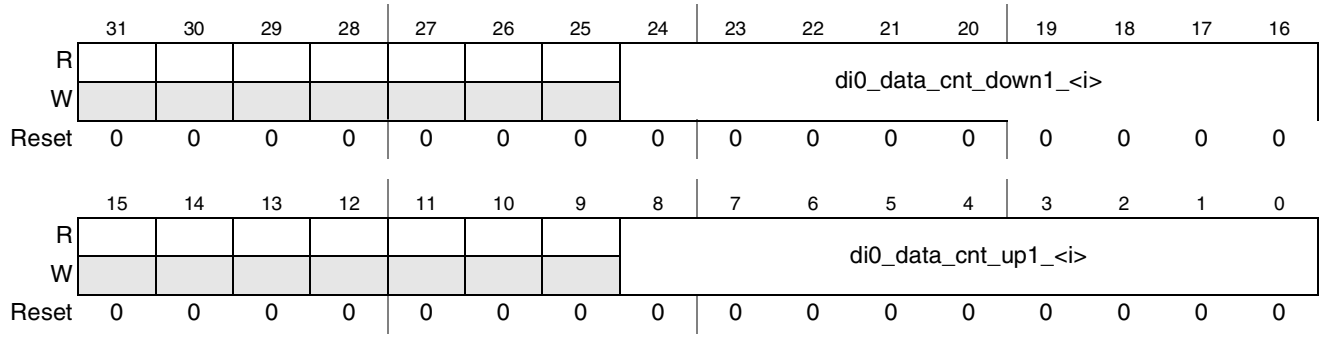


Figure 42-225. DI0 Data Wave Set 1 <i> Registers (DI0_DW_SET1_<i>)

Table 42-226. DI0_DW_SET1_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_data_cnt_down1_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>
15–9	Reserved.
8–0 di0_data_cnt_up1_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>

42.2.3.9.26 DI0 Data Wave Set 2 <i> Registers (DI0_DW_SET2_<i>)

Address 0xBASE+0xE0400E8 (DI0_DW_SET2_<i>) Access: User read/write
offset 4
range 0 .. 11

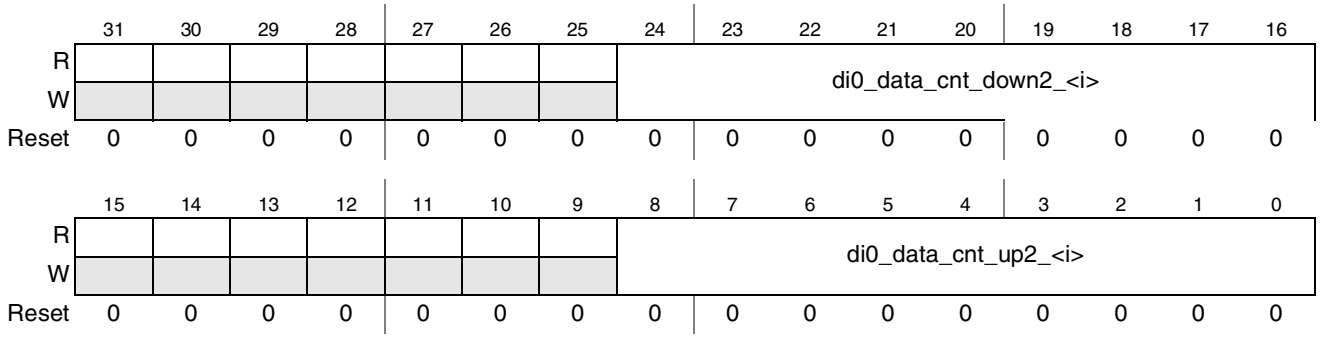


Figure 42-226. DI0 Data Wave Set 2 <i> Registers (DI0_DW_SET2_<i>)

Table 42-227. DI0_DW_SET2_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_data_cnt_down2_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>
15–9	Reserved.
8–0 di0_data_cnt_up2_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>

42.2.3.9.27 DI0 Data Wave Set 3 <i> Registers (DI0_DW_SET3_<i>)

Address 0xBASE+0xE040118 (DI0_DW_SET3_<i>) Access: User read/write
offset 4
range 0 .. 11

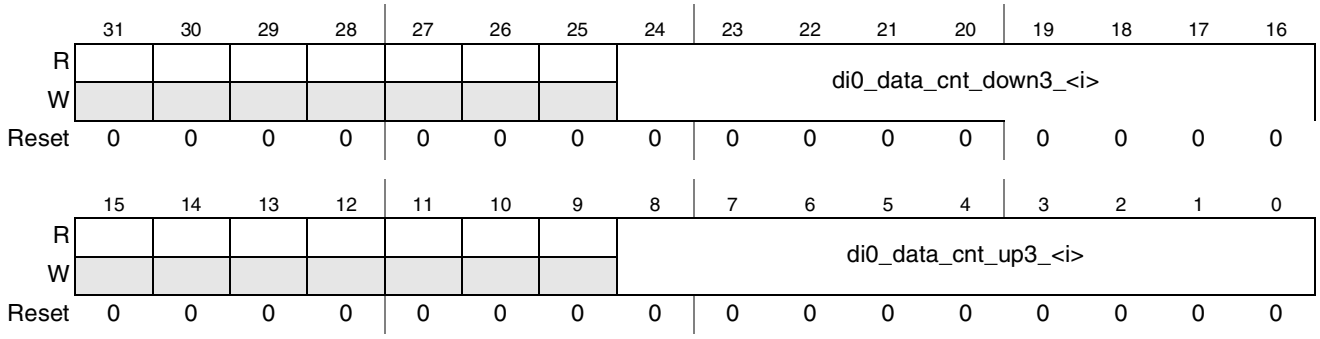


Figure 42-227. DI0 Data Wave Set 3 <i> Registers (DI0_DW_SET3_<i>)

Table 42-228. DI0_DW_SET3_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di0_data_cnt_down3_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>
15–9	Reserved.
8–0 di0_data_cnt_up3_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di0_pt_*_<i>

42.2.3.9.28 DI0 Step Repeat <i> Registers (DI0_STP_REP_<i>)

Address 0xBASE+0xE040148 (DI0_STP_REP_<i>)
offset 4
range 1 .. 4

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	di0_step_repeat_<2*i>											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					di0_step_repeat_<2*i-1>											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-228. DI0 Step Repeat <i> Registers (DI0_STP_REP_<i>)

Table 42-229. DI0_STP_REP_<i>Field Descriptions

Field	Description
27-16 11-0 di0_step_repeat_<i>	Step Repeat <i> This fields defines the amount of repetitions that will be performed by the counter <i>

42.2.3.9.29 DI0 Step Repeat 9 Registers (DI0_STP_REP_9)

Address 0xBASE+0xE040158 (DI0_STP_REP_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	di0_step_repeat_9											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-229. DI0 Step Repeat 9 Registers (DI0_STP_REP_9)

Table 42-230. DI0_STP_REP_9 Field Descriptions

Field	Description
27-16 11-0 di0_step_repeat_9	Step Repeat 9 This fields defines the amount of repetitions that will be performed by the counter 9

42.2.3.9.30 DI0 Serial Display Control Register (DI0_SER_CONF)

Address 0xBASE+0xE04015C (DI0_SER_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DI0_SERIAL_LLA_PNTR_RS_R_1				DI0_SERIAL_LLA_PNTR_RS_R_0				DI0_SERIAL_LLA_PNTR_RS_W_1				DI0_SERIAL_LLA_PNTR_RS_W_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI0_SERIAL_LATCH								0	0	DI0_LLA_SER_ACCESS	DI0_SER_CLK_POLARITY	DI0_SERIAL_DATA_POLARITY	DI0_SERIAL_RS_POLARITY	DI0_SERIAL_CS_POLARITY	DI0_WAIT4SERIAL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-230. DI0 Serial Display Control Register (DI0_SER_CONF)

Table 42-231. DI0_SER_CONF Field Descriptions

Field	Description
31–28 DI0_SERIAL_LLA_PNTR_RS_R_1	RS 3 waveform pointer for read low level access This pointer defines which waveform set will be chosen when the read low level access is targeted to RS group 1. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
27–24 DI0_SERIAL_LLA_PNTR_RS_R_0	RS 2 waveform pointer for read low level access This pointer defines which waveform set will be chosen when the read low level access is targeted to RS group 0. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
23–20 DI0_SERIAL_LLA_PNTR_RS_W_1	RS 1 waveform pointer for write low level access This pointer defines which waveform set will be chosen when the low level write access is targeted to RS group 1. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
19–16 DI0_SERIAL_LLA_PNTR_RS_W_0	RS 0 waveform pointer for write low level access This pointer defines which waveform set will be chosen when the low level write access is targeted to RS group 0. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
15–8 DI0_SERIAL_LATCH	DI0 Serial Latch This field defines how many cycles to insert between serial read accesses start to data sampling in the IPUv3EX.
7–6	Reserved

Table 42-231. DI0_SER_CONF Field Descriptions (continued)

Field	Description
5 DI0_LLA_SER_ACCESS	Direct Low Level Access to Serial display 1MCU access is performed via a direct path to the serial display in LLA mode, in this mode only the MCU in LLA mode can access the serial port 0 MCU access to the serial display port is not done directly, hence other source are allowed to access the serial port. The arbitration is done automatically
4 DI0_SER_CLK_POLARITY	Serial Clock Polarity The output polarity of the SER_CLK pin 1 - The clock is inverted 0 - The clock is not inverted
3 DI0_SERIAL_DATA_POLARITY	Serial Data Polarity The output polarity of the SER_DATA pin 1 - The data is inverted 0 - The data is not inverted
2 DI0_SERIAL_RS_POLARITY	Serial RS Polarity The output polarity of the SER_RS pin 1 - The RS is inverted 0 - The RS is not inverted
1 DI0_SERIAL_CS_POLARITY	Serial Chip Select Polarity The output polarity of the SER_CS pin 1 - The CS is inverted 0 - The CS is not inverted
0 DI0_WAIT4SERIAL	Wait for Serial When the parallel display share pins with the serial port. Accessing the parallel port is not allowed till the serial port completes its access. 1 The parallel port should wait to the serial port as the pins are shared 0 The parallel port should not wait to the serial port as the pins are not shared

42.2.3.9.31 DI0 Special Signals Control Register (DI0_SSC)

Address 0xBASE+0xE040160 (DI0_SSC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	DI0_PIN17_ERM	DI0_PIN16_ERM	DI0_PIN15_ERM	DI0_PIN14_ERM	DI0_PIN13_ERM	DI0_PIN12_ERM	DI0_PIN11_ERM	DI0_CS_ERM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	DI0_WAIT_ON	0	DI0_BYTE_EN_RD_IN	DI0_BYTE_EN_PNTR		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-231. DI0 Special Signals Control Register (DI0_SSC)

Table 42-232. DI0_SSC Field Descriptions

Field	Description
31–24	Reserved.
23 DI0_PIN17_ERM	DI0 PIN17 error recovery mode. This bit defines the error recovery mode of the PIN17 pin. 1 - The PIN17 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN17 pin following a display error detection
22 DI0_PIN16_ERM	DI0 PIN16 error recovery mode. This bit defines the error recovery mode of the PIN16 pin. 1 - The PIN16 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN16 pin following a display error detection
21 DI0_PIN15_ERM	DI0 PIN15 error recovery mode. This bit defines the error recovery mode of the PIN15 pin. 1 - The PIN15 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN15 pin following a display error detection
20 DI0_PIN14_ERM	DI0 PIN14 error recovery mode. This bit defines the error recovery mode of the PIN14 pin. 1 - The PIN14 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN14 pin following a display error detection
19 DI0_PIN13_ERM	DI0 PIN13 error recovery mode. This bit defines the error recovery mode of the PIN13 pin. 1 - The PIN13 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN13 pin following a display error detection

Table 42-232. DI0_SSC Field Descriptions (continued)

Field	Description
18 DI0_PIN12_ERM	DI0 PIN12 error recovery mode. This bit defines the error recovery mode of the PIN12 pin. 1 - The PIN12 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN12 pin following a display error detection
17 DI0_PIN11_ERM	DI0 PIN11 error recovery mode. This bit defines the error recovery mode of the PIN11 pin. 1 - The PIN11 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN11 pin following a display error detection
16 DI0_CS_ERM	DI0 GLUELOGIC error recovery mode. This bit defines the error recovery mode of the GLUELOGIC (see Table 42-425). 1 - The GLUELOGIC is release in case of a synchronous display error. The release will be done on the next VSYNC 0 - Nothing is done to the GLUELOGIC following a display error detection
15–6	Reserved.
5 DI0_WAIT_ON	Wait On This field defines the DC's response to WAIT signal 1 - The DC holds the flow as long as WAIT is asserted 0 - The DC continues the flow regardless the WAIT signal
4	Reserved
3 DI0_BYTE_EN_RD_IN	Byte Enable Read In This bit selects the source of the byte enable pins 1 The write byte enable signals are routed via bits [17:16] of the display's data, The read byte enable signals are routed via bits [19:18 of the display's data 0 The byte enable signals are routed via bits [17:16] of the display's data for both read and write
2-0 DI0_BYTE_EN_PNTR	Byte Enable Pointer This pointer selects the pin asserted along with the byte enables signals 000 wave form of byte enable as pin_11 001 wave form of byte enable as pin_12 ... 111 wave form of byte enable as suitable CS pin

42.2.3.9.32 DI0 Polarity Register (DI0_POL)

Address 0xBASE+0xE040164 (DI0_POL) Access: User read/write

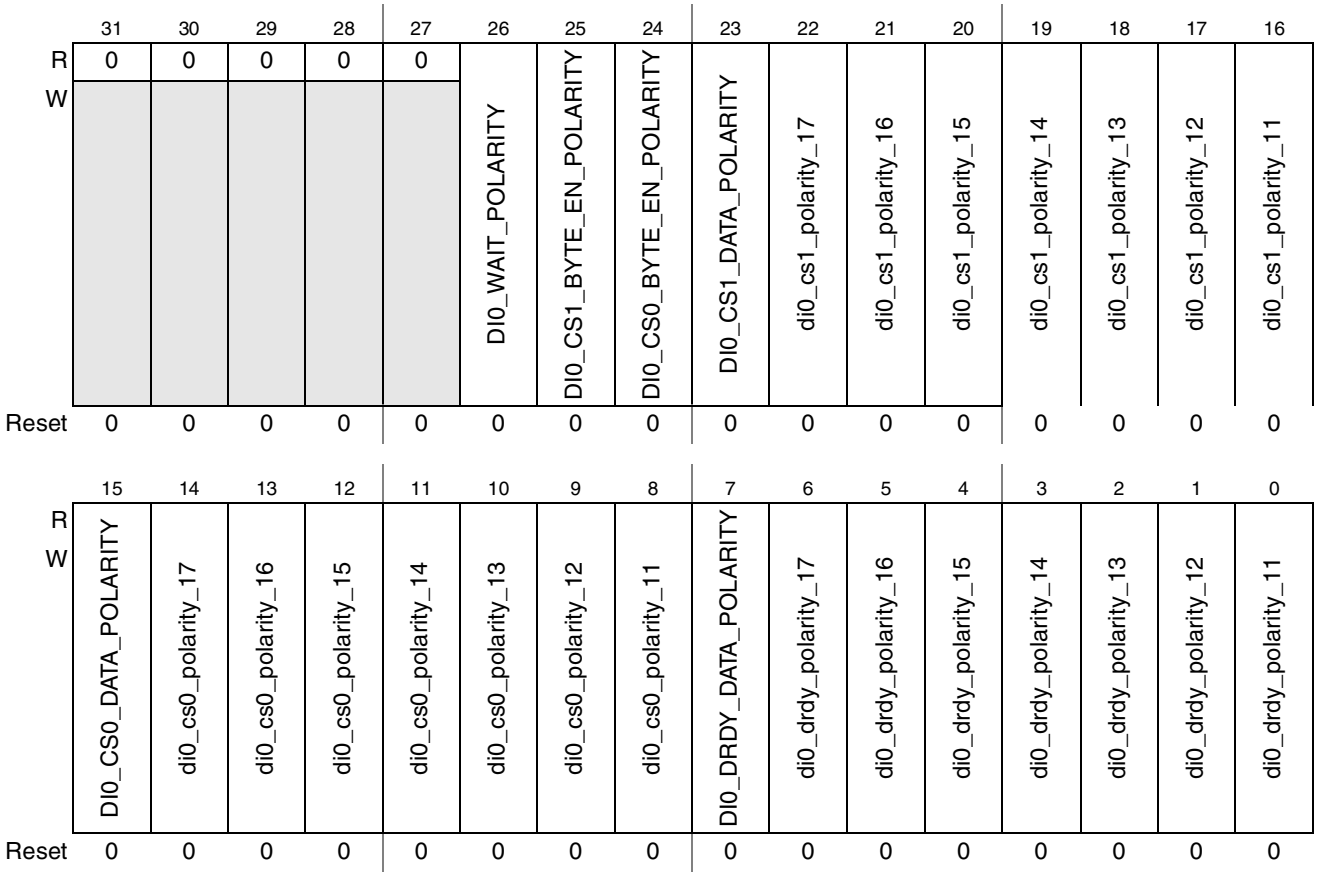


Figure 42-232. DI0 Polarity Register (DI0_POL)

Table 42-233. DI0_POL Field Descriptions

Field	Description
31-27	Reserved.
26 DI0_WAIT_POLARITY	WAIT polarity This bit defines the polarity of the wait signal input coming from the displa1 1 - active high 0 active low
25 DI0_CS1_BYTE_EN_POLARITY	Byte Enable associated with CS1 polarity This bit defines the polarity of the byte enable signals to the display 1 - active high 0 active low
24 DI0_CS0_BYTE_EN_POLARITY	Byte Enable associated with CS0 polarity This bit defines the polarity of the byte enable signals to the display 1 - active high 0 active low

Table 42-233. DI0_POL Field Descriptions (continued)

Field	Description
23 DI0_CS1_DATA_POLARITY	Data Polarity associated with CS1
22-16 di0_cs1_polarity_[17:11]	DI0 output pin's polarity for CS1 This bits define the polarity of each of the DI's outputs when CS1 is asserted 1 The output pin is active high 0 The output pin is active low
15 DI0_CS0_DATA_POLARITY	Data Polarity associated with CS0
14-8 di0_cs0_polarity_[17:11]	DI0 output pin's polarity for CS1 This bits define the polarity of each of the DI's outputs when CS1 is asserted 1 The output pin is active high 0 The output pin is active low
7 DI0_DRDY_DATA_POLARITY	Data Polarity associated with DRDY
6-0 di0_drdy_polarity_[17:11]	DI0 output dynamic pin's polarity for synchronous access This bits define the polarity of each of the DI's outputs when synchronous display access is asserted 1 The output pin is active high 0 The output pin is active low The pins' default polarity is the same as defined in the di0_drdy_polarity_[17:11] bits

42.2.3.9.33 DI0 Active Window 0 Register (DI0_AW0)

Address 0xBASE+0xE040168 (DI0_AW0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI0_AW_TRIG_SEL				DI0_AW_HEND											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI0_AW_HCOUNT_SEL				DI0_AW_HSTART											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-233. DI0 Active Window 0 Register (DI0_AW0)

Table 42-234. DI0_AW0 Field Descriptions

Field	Description
31-28 DI0_AW_TRIG_SEL	This field selects the trigger for sending data during the display's active window 000 - disabled 001 - The trigger is the same trigger that triggers the displays clock. 010 - The trigger is counter #1 011 - The trigger is counter #2 100 - The trigger is counter #3 101 - The trigger is counter #4 110 - The trigger is counter #5 111 The trigger is always on.
27-16 DI0_AW_HEND	This field defines the horizontal end of the active window
15-12 DI0_AW_HCOUNT_SEL	GM: This field selects the counter that counts the horizontal position of the display's active window 0000 - disabled 0001 - reserved 0010 - The counter is counter #1 0011 - The counter is counter #2 0100 - The counter is counter #3 0101 - The counter is counter #4 0110 - The counter is counter #5 ..... 1001 - The counter is counter #8
11-0 DI0_AW_HSTART	This field defines the horizontal start of the active window DI0_AW_HSTART < DI0_AW_HEND

42.2.3.9.34 DI0 Active Window 1 Register (DI0_AW1)

Address 0xBASE+0xE04016C (DI0_AW1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	DI0_AW_VEND											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI0_AW_VCOUNT_SEL				DI0_AW_VSTART											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-234. DI0 Active Window 1 Register (DI0_AW1)

Table 42-235. DI0_AW1 Field Descriptions

Field	Description
31-28	Reserved
27-16 DI0_AW_VEND	This field defines the vertical end of the active window
15-12 DI0_AW_VCOUNT_SEL	This field selects the counter that counts the vertical position of the display's active window 0000 - disabled 0001 - reserved 0010 - The counter is counter #1 0011 - The counter is counter #2 0100 - The counter is counter #3 0101 - The counter is counter #4 0110 - The counter is counter #5 ..... 1001 - The counter is counter #8
11-0 DI0_AW_VSTART	This field defines the vertical start of the active window DI0_AW_VSTART < DI0_AW_VEND

42.2.3.9.35 DI0 Screen Configuration Register (DI0_SCR_CONF)

Address 0xBASE+0xE040170 (DI0_SCR_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	DI0_SCREEN_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-235. DI0 Screen Configuration Register (DI0_SCR_CONF)

Table 42-236. DI0_SCR_CONF Field Descriptions

Field	Description
31–12	Reserved.
11–0 DI0_SCREEN_HEIGHT	This field defines the number of display rows (Number_of_ROWS = DI0_SCREEN_HEIGHT+1) This field is used for VSYNC calculation and for anti-tearing

42.2.3.9.36 DI0 Status Register (DI0_STAT)

Address 0xBASE+0xE040174 (DI0_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	DI0_CNTR_FIFO_FULL	DI0_CNTR_FIFO_EMPTY	DI0_READ_FIFO_FULL	DI0_READ_FIFO_EMPTY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Figure 42-236. DI0 Status Register (DI0_STAT)

Table 42-237. DI0_STAT Field Descriptions

Field	Description
31-4	Reserved
3 DI0_CNTR_FIFO_FULL	
2 DI0_READ_CNTR_EMPTY	
1 DI0_READ_FIFO_FULL	
0 DI0_READ_FIFO_EMPTY	

42.2.3.10 DI1 Registers

42.2.3.10.1 DI1 General Register (DI1_GENERAL)

Address 0xBASE+0xE048000 (DI1_GENERAL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	di1_pin8_pin15_sel	di1_disp_y_sel			DI1_CLOCK_STOP_MODE				DI1_DISP_CLOCK_INIT	di1_mask_sel	di1_vsync_ext	di1_clk_ext	DI1_WATCHDOG_MODE		di1_polarity_disp_clk	0
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	di1_sync_count_sel				di1_err_treatment	di1_erm_vsync_sel	di1_polarity_cs1	di1_polarity_cs0	di1_polarity_8	di1_polarity_7	di1_polarity_6	di1_polarity_5	di1_polarity_4	di1_polarity_3	di1_polarity_2	di1_polarity_1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-237. DI1 General Register (DI1_GENERAL)

Table 42-238. DI1_GENERAL Field Descriptions

Field	Description
31 di1_pin8_pin15_sel	This bit routes PIN8 over PIN15 1 - PIN8 is routed to PIN15, PIN8 is also routed to PIN8 0 - PIN15 is routed to PIN15, PIN8 is routed to PIN8\
30–28 di1_disp_y_sel	DI1 Display Vertical coordinate (Y) select. This field defines which one of the 8 counters will be used as a display's line counter. 000 counter #1 is selected ... 111 counter #8 is selected

Table 42-238. DI1_GENERAL Field Descriptions (continued)

Field	Description
27–24 DI1_CLOCK_STOP_MODE	DI clock stop mode When performing a clock change. The DI stops the clock to the display. These field defines when the clock will be stopped 0000 - stop at the next edge of the display clock 0001-1001 stop at the next event of one of the counters (counter #1 to counter #9) 1100 - stop at EOL (end of a line), but if stop request is during blanking interval, stop now 1101 - stop at EOF (end of a frame), but if stop request is during blanking interval, stop now 1110 - stop at EOL (end of a line), but if stop request is during blanking interval, stop at the end of the next line 1111 - stop at EOF (end of a frame), but if stop request is during blanking interval, stop at the end of the next frame Stopping at EOL/EOF is supported for the case where the data is coming from the IDMAC (DMA access). In case that only direct accesses is performed, the user should set this field to 0000
23 DI1_DISP_CLOCK_INIT	Display clock's initial mode For synchronization error conditions the display clock can be stopped on the next VSYNC 1 - The display's clock is running after the next VSYNC (indicating new frame) 0 - The display's clock is stopped after the next VSYNC (indicating new frame)
22 di1_mask_sel	DI1 Mask select. IPP_PIN_2 output of the DI that functions as MASK signal can come from 2 sources: counter #2 or extracted from the MASK data coming from the memory. 1 IPP_PIN_2 is coming from extracted MASK data coming from the memory 0 IPP_PIN_2 is coming from counter #2
21 di1_vsync_ext	DI1 External VSYNC. This bit selects the source of the VSYNC signal 1 External to the IPUv3EX 0 Internally generated by the IPUv3EX
20 di1_clk_ext	DI1 External Clock. This bit selects the source of the DI's clock 1 The source of the clock is external to the IPUv3EX 0 The clock is internally generated by the IPUv3EX
19-18 DI1_WATCHDOG_MODE	DI1 watchdog mode In case of a display error where the DI clock is stopped (defined at di0_err_treatment). An internal watchdog counts DI clocks. If this timer reached its pre defined value the DI will skip the current frame and restart on the frame. This 2 bits define the number of DI clock cycles that the timer counts. 00 The timer counts 4 DI cycles 01 The timer counts 16 DI cycles 10 The timer counts 64 DI cycles 11 The timer counts 128 DI cycles
17 di1_polarity_disp_clk	DI1 Output Clock's polarity This bits define the polarity of the DI's clock. 1 The output clock is active high 0 The output clock is active low
16	Reserved
15–12 di1_sync_count_sel	For synchronous flow error: selects synchronous flow synchronization counter in DI:

Table 42-238. DI1_GENERAL Field Descriptions (continued)

Field	Description
11 di1_err_treatment	In case of synchronous flow error there are 2 ways to handle the display 1 to wait (i.e. stop clock) 0 Drive the last component
10 di1_erm_vsync_sel	DI1 error recovery module's VSYNC source select The error recovery module detect a case where the DI's VSYNC is asserted before the EOF. This bit selects the source of the VSYNC signal monitored by this mechanism. 1- vsync_post - an internal VSYNC signal asserted 2 lines after the DI's VSYNC 0 - vsync_pre - an internal VSYNC signal asserted 2 lines before the DI's VSYNC
9 di1_polarity_cs1	DI1 Chip Select's 1 polarity This bits define the polarity of the DI's CS1. 1 The CS1 is active high 0 The CS1 is active low
8 di1_polarity_cs0	DI1 Chip Select's 0 polarity This bits define the polarity of the DI's CS0. 1 The CS0 is active high 0 The CS0 is active low
7-0 di1_polarity_<i+1>	DI1 output pin's polarity This bits define the polarity of each of the DI's outputs. 1 The output pin is active high 0 The output pin is active low

42.2.3.10.2 DI1 Base Sync Clock Gen 0 Register (DI1_BS_CLKGEN0)

Address 0xBASE+0xE048004 (DI1_BS_CLKGEN0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di1_disp_clk_offset								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	di1_disp_clk_period											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-238. DI0 Base Sync Clock Gen 0 Register (DI0_BS_CLKGEN0)

Table 42-239. DI1_BS_CLKGEN0 Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_disp_clk_offset	DI1 Display Clock Offset The DI has the ability to delay the display's clock This field defines the amount of IPUv3EX's clock cycles added as delay on this clock.

Table 42-239. DI1_BS_CLKGEN0 Field Descriptions (continued)

Field	Description
15–12	Reserved.
11–0 di1_disp_clk_period	DI1 Display Clock Period This field defines the Display interface clock period for display write access. This parameter contains an integer part (bits 11:4) and a fractional part (bits 3:0). It defines a fractional division ratio of the HSP_CLK clock for generation of the display's interface clock.

42.2.3.10.3 DI1 Base Sync Clock Gen 1 Register (DI1_BS_CLKGEN1)

Address 0xBASE+0xE048008 (DI1_BS_CLKGEN1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di1_disp_clk_down								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	di1_disp_clk_up								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-239. DI0 Base Sync Clock Gen 1 Register (DI0_BS_CLKGEN1)

Table 42-240. DI1_BS_CLKGEN1 Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_disp_clk_down	DI1 display clock falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is a time interval between display's access start point and display 's interface clock falling edge.
15–9	Reserved.
8–0 di1_disp_clk_up	DI1 display clock rising edge position This parameter contains an integer part (bits 8:1) and a fractional part (bit 0). The position value is a time interval between display's access start point and display 's interface clock rising edge.

42.2.3.10.4 DI1 Sync Wave Gen 1 Register 0 (DI1_SW_GEN0_1)

Address 0xBASE+0xE04800C (DI1_SW_GEN0_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_1												di1_run_resolution_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_1												di1_offset_resolution_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-240. DI0 Sync Wave Gen 1 Register 0 (DI0_SW_GEN0_1)

Table 42-241. DI1_SW_GEN0_1 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_1	DI1 counter #1 pre defined value This fields defines the counter #1 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_1 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_1	DI1 counter #1 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - NA 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di1_offset_value_1	DI1 counter #1 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_1	DI1 counter #1 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - NA 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.10.5 DI1 Sync Wave Gen 2 Register 0 (DI1_SW_GEN0_2)

Address 0xBASE+0xE048010 (DI1_SW_GEN0_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_2												di1_run_resolution_2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_2												di1_offset_resolution_2		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-241. DI0 Sync Wave Gen 2 Register 0 (DI0_SW_GEN0_2)

Table 42-242. DI1_SW_GEN0_2 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_2	DI1 counter #2 pre defined value This fields defines the counter #2 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_2 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_2	DI1 counter #2 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di1_offset_value_2	DI1 counter #2 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_2	DI1 counter #2 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - NA 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.10.6 DI1 Sync Wave Gen 3 Register 0 (DI1_SW_GEN0_3)

Address 0xBASE+0xE048014 (DI1_SW_GEN0_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_3												di1_run_resolution_3		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_3												di1_offset_resolution_3		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-242. DI0 Sync Wave Gen 3 Register 0 (DI0_SW_GEN0_3)

Table 42-243. DI1_SW_GEN0_3 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_3	DI1 counter #3 pre defined value This fields defines the counter #3 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_3 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_3	DI1 counter #3 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
14–3 di1_offset_value_3	DI1 counter #3 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_3	DI1 counter #3 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - NA 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.10.7 DI1 Sync Wave Gen 4 Register 0 (DI1_SW_GEN0_4)

Address 0xBASE+0xE048018 (DI1_SW_GEN0_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_4												di1_run_resolution_4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_4												di1_offset_resolution_4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-243. DI0 Sync Wave Gen 4 Register 0 (DI0_SW_GEN0_4)

Table 42-244. DI1_SW_GEN0_4 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_4	DI1 counter #4 pre defined value This fields defines the counter #4 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_4 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_4	DI1 counter #4 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

Table 42-244. DI1_SW_GEN0_4 Field Descriptions (continued)

Field	Description
14–3 di1_offset_value_4	DI1 counter #4 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_4	DI1 counter #4 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

42.2.3.10.8 DI1 Sync Wave Gen 5 Register 0 (DI1_SW_GEN0_5)

Address 0xBASE+0xE04801C (DI1_SW_GEN0_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_5												di1_run_resolution_5		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_5												di1_offset_resolution_5		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-244. DI0 Sync Wave Gen 5 Register 0 (DI0_SW_GEN0_5)

Table 42-245. DI1_SW_GEN0_5 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_5	DI1 counter #5 pre defined value This fields defines the counter #5 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_5 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.

Table 42-245. DI1_SW_GEN0_5 Field Descriptions (continued)

Field	Description
18–16 di1_run_resolution_5	DI1 counter #5 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.
14–3 di1_offset_value_5	DI1 counter #5 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_5	DI1 counter #5 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 -The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.

42.2.3.10.9 DI1 Sync Wave Gen 6 Register 0 (DI1_SW_GEN0_6)

Address 0xBASE+0xE048020 (DI1_SW_GEN0_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_6												di1_run_resolution_6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_6												di1_offset_resolution_6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-245. DI0 Sync Wave Gen 6 Register 0 (DI0_SW_GEN0_6)

Table 42-246. DI1_SW_GEN0_6 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_6	DI1 counter #6 pre defined value This fields defines the counter #6 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_6 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_6	DI1 counter #6 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di1_offset_value_6	DI1 counter #6 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_6	DI1 counter #6 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.10.10 DI1 Sync Wave Gen 7 Register 0 (DI1_SW_GEN0_7)

Address 0xBASE+0xE048024 (DI1_SW_GEN0_7) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_7												di1_run_resolution_7		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_7												di1_offset_resolution_7		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-246. DI0 Sync Wave Gen 7 Register 0 (DI0_SW_GEN0_7)

Table 42-247. DI1_SW_GEN0_7 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_7	DI1 counter #7 pre defined value This fields defines the counter #7 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_7 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_7	DI1 counter #1 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di1_offset_value_7	DI1 counter #7 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_1	DI1 counter #7 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.10.11 DI1 Sync Wave Gen 8 Register 0 (DI1_SW_GEN0_8)

Address 0xBASE+0xE048028 (DI1_SW_GEN0_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_8												di1_run_resolution_8		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_8												di1_offset_resolution_8		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-247. DI0 Sync Wave Gen 8 Register 0 (DI0_SW_GEN0_8)

Table 42-248. DI1_SW_GEN0_8 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_8	DI1 counter #8 pre defined value This fields defines the counter #8 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_8 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_8	DI1 counter #8 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
14–3 di1_offset_value_8	DI1 counter #8 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_8	DI1 counter #8 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.10.12 DI1 Sync Wave Gen 9 Register 0 (DI1_SW_GEN0_9)

Address 0xBASE+0xE04802C (DI1_SW_GEN0_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_run_value_m1_9												di1_run_resolution_9		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_offset_value_9												di1_offset_resolution_9		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-248. DI1 Sync Wave Gen 9 Register 0 (DI1_SW_GEN0_9)

Table 42-249. DI1_SW_GEN0_9 Field Descriptions

Field	Description
31	Reserved.
30–19 di1_run_value_m1_9	DI1 counter #9 pre defined value This fields defines the counter #9 pre defines value. When the counter is auto reload mode (di1_cnt_auto_reload_9 bit is set), the counter automatically restarts. otherwise the counter restarts for the amount of time defined on the corresponding di1_step_repeat field.
18–16 di1_run_resolution_9	DI1 counter #9 Run Resolution This field defines the trigger causing the counter to increment. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

Table 42-249. DI1_SW_GEN0_9 Field Descriptions (continued)

Field	Description
14–3 di1_offset_value_9	DI1 counter #9 offset value The counter can start counting after a pre defined delay This field defines the amount of cycles that the counter will be delayed by
2–0 di1_offset_resolution_9	DI1 counter #9 offset Resolution This field defines the trigger causing the offset counter to increment 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

42.2.3.10.13 DI1 Sync Wave 1 Gen Register 1 (DI1_SW_GEN1_1)

Address 0xBASE+0xE048030 (DI1_SW_GEN1_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_1		di1_cnt_auto_reload_1	di1_cnt_clr_sel_1				di1_cnt_down_1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_1			di1_cnt_polarity_clr_sel_1				di1_cnt_up_1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-249. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-250. DI1_SW_GEN1_1 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_1	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_1	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_1 field
27–25 di1_cnt_clr_sel_1	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di1_cnt_down_1	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_1	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - Reserved 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

Table 42-250. DI1_SW_GEN1_1 Field Descriptions (continued)

Field	Description
11–9 di1_cnt_polarity_clr_sel_1	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Reserved 011 - Reserved 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di1_cnt_up_1	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.14 DI1 Sync Wave 2 Gen Register 1 (DI1_SW_GEN1_2)

Address 0xBASE+0xE048034 (DI1_SW_GEN1_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_2			di1_cnt_auto_reload_2	di1_cnt_clr_sel_2			di1_cnt_down_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_2			di1_cnt_polarity_clr_sel_2				di1_cnt_up_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-250. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-251. DI1_SW_GEN1_2 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_2	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_2	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field

Table 42-251. DI1_SW_GEN1_2 Field Descriptions (continued)

Field	Description
27–25 di1_cnt_clr_sel_2	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di1_cnt_down_2	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_2	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - Reserved 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–9 di1_cnt_polarity_clr_sel_2	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Reserved 011 - Reserved 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di1_cnt_up_2	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.15 DI1 Sync Wave 3 Gen Register 1 (DI1_SW_GEN1_3)

Address 0xBASE+0xE048038 (DI1_SW_GEN1_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_3		di1_cnt_auto_reload_3	di1_cnt_clr_sel_3			di1_cnt_down_3								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_3			di1_cnt_polarity_clr_sel_3			di1_cnt_up_3								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-251. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-252. DI1_SW_GEN1_3 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_3	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_3	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field
27–25 di1_cnt_clr_sel_3	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di1_cnt_down_3	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved

Table 42-252. DI1_SW_GEN1_3 Field Descriptions (continued)

Field	Description
14–12 di1_cnt_polarity_trigger_sel_3	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - Reserved 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
11–9 di1_cnt_polarity_clr_sel_3	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Reserved 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di1_cnt_up_3	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.16 DI1 Sync Wave 4 Gen Register 1 (DI1_SW_GEN1_4)

Address 0xBASE+0xE04803C (DI1_SW_GEN1_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_4		di1_cnt_auto_reload_4	di1_cnt_clr_sel_4			di1_cnt_down_4								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_4			di1_cnt_polarity_clr_sel_4			di1_cnt_up_4								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-252. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-253. DI1_SW_GEN1_4 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_4	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_4	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field
27–25 di1_cnt_clr_sel_4	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.
24–16 di1_cnt_down_4	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_4	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - CSI VSYNC. The VSYNC is a trigger coming from one of the CSI's according to the CSI_VSYNC_DEST bit. 110 - External VSYNC 111 Counter is always on.

Table 42-253. DI1_SW_GEN1_4 Field Descriptions (continued)

Field	Description
11–9 di1_cnt_polarity_clr_sel_4	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Reserved 110 - Reserved 111 - Reserved
8–0 di1_cnt_up_4	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.17 DI1 Sync Wave 5 Gen Register 1 (DI1_SW_GEN1_5)

Address 0xBASE+0xE048040 (DI1_SW_GEN1_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_5		di1_cnt_auto_reload_5	di1_cnt_clr_sel_5				di1_cnt_down_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_5			di1_cnt_polarity_clr_sel_5				di1_cnt_up_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-253. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-254. DI1_SW_GEN1_5 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_5	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_5	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field

Table 42-254. DI1_SW_GEN1_5 Field Descriptions (continued)

Field	Description
27–25 di1_cnt_clr_sel_5	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.
24–16 di1_cnt_down_5	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_5	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - External VSYNC 111 Counter is always on.
11–9 di1_cnt_polarity_clr_sel_5	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Reserved 111 - Reserved
8–0 di1_cnt_up_5	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.18 DI1 Sync Wave 6 Gen Register 1 (DI1_SW_GEN1_6)

Address 0xBASE+0xE048044 (DI1_SW_GEN1_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_6		di1_cnt_auto_reload_6	di1_cnt_clr_sel_6			di1_cnt_down_6								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_6			di1_cnt_polarity_clr_sel_6			di1_cnt_up_6								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-254. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-255. DI1_SW_GEN1_6 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_6	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_6	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field
27–25 di1_cnt_clr_sel_6	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di1_cnt_down_6	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved

Table 42-255. DI1_SW_GEN1_6 Field Descriptions (continued)

Field	Description
14–12 di1_cnt_polarity_trigger_sel_6	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
11–9 di1_cnt_polarity_clr_sel_6	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Reserved
8–0 di1_cnt_up_6	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.19 DI1 Sync Wave 7 Gen Register 1 (DI1_SW_GEN1_7)

Address 0xBASE+0xE048048 (DI1_SW_GEN1_7) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_7		di1_cnt_auto_reload_7	di1_cnt_clr_sel_7			di1_cnt_down_7								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_7			di1_cnt_polarity_clr_sel_7			di1_cnt_up_7								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-255. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-256. DI1_SW_GEN1_7 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_7	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_7	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field
27–25 di1_cnt_clr_sel_7	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di1_cnt_down_7	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_7	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

Table 42-256. DI1_SW_GEN1_7 Field Descriptions (continued)

Field	Description
11–9 di1_cnt_polarity_clr_sel_7	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Output is inverted if the output of counter #6 is set
8–0 di1_cnt_up_7	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.20 DI1 Sync Wave 8 Gen Register 1 (DI1_SW_GEN1_8)

Address 0xBASE+0xE04804C (DI1_SW_GEN1_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	di1_cnt_polarity_gen_en_8			di1_cnt_auto_reload_8	di1_cnt_clr_sel_8			di1_cnt_down_8							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	di1_cnt_polarity_trigger_sel_8			di1_cnt_polarity_clr_sel_8				di1_cnt_up_8							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-256. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-257. DI1_SW_GEN1_8 Field Descriptions

Field	Description
31	Reserved.
30–29 di1_cnt_polarity_gen_en_8	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_8	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field

Table 42-257. DI1_SW_GEN1_8 Field Descriptions (continued)

Field	Description
27–25 di1_cnt_clr_sel_8	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
24–16 di1_cnt_down_8	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15	Reserved
14–12 di1_cnt_polarity_trigger_sel_8	DI1 Counter's toggling trigger select This field selects the counter's trigger causing the output to toggle 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.
11–9 di1_cnt_polarity_clr_sel_8	DI1 counter's polarity Clear select This field selects the input to the counter telling the counter wether to invert the output 000 - Output is always inverted 001 - Output is kept the same (no inversion) 010 - Output is inverted if the output of counter #1 is set 011 - Output is inverted if the output of counter #2 is set 100 - Output is inverted if the output of counter #3 is set 101 - Output is inverted if the output of counter #4 is set 110 - Output is inverted if the output of counter #5 is set 111 - Output is inverted if the output of counter #6 is set
8–0 di1_cnt_up_8	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.21 DI1 Sync Wave 9 Gen Register 1 (DI1_SW_GEN1_9)

Address 0xBASE+0xE048050 (DI1_SW_GEN1_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	di1_gentime_sel_9			di1_cnt_auto_reload_9	di1_cnt_clr_sel_9			di1_cnt_down_9								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	di1_tag_sel_9	0	0	0	0	0	0	di1_cnt_up_9								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-257. DI0 Sync Wave 1 Gen Register 1 (DI0_SW_GEN1_1)

Table 42-258. DI1_SW_GEN1_9 Field Descriptions

Field	Description
31-29 di1_gentime_sel_9	Counter #9 main waveform select This field defines the counter that counter #9's auxiliary waveform will be attached too. 000 - Counter #9's waveform is attached to counter #1's waveform 001 - Counter #9's waveform is attached to counter #2's waveform 010 - Counter #9's waveform is attached to counter #3's waveform 011 - Counter #9's waveform is attached to counter #4's waveform 100 - Counter #9's waveform is attached to counter #5's waveform 101 - Counter #9's waveform is attached to counter #6's waveform 110 - Counter #9's waveform is attached to counter #7's waveform 111 - Counter #9's waveform is attached to counter #8's waveform
30–29 di1_cnt_polarity_gen_en_9	DI1 Counter polarity generator enable The counter's output polarity can be changed on the fly. 00 Dynamic polarity change is disabled 01 The counters UP and DOWN value are calculated according to the trigger selected by cnt_polarity_trigger_sel field. When selecting this mode the UP and DOWN values has to be a multiple of 2. 10 Dynamic polarity change is enabled 11 Outputs polarity is dynamically changed every time the counter reaches its pre defined value
28 di1_cnt_auto_reload_9	Counter auto reload mode 1 - The counter will automatically be reloaded forever, ignoring the value of the di1_step_repeat_<i> field 0 - The counter will not be automatically reloaded, It will be reloaded for the amount of repeat times defined on the di1_step_repeat_<i> field
27–25 di1_cnt_clr_sel_9	Counter Clear select This field defines the source of the signals that clears the counter. 000 - Counter is disabled 001 - The counter is triggered by the same trigger that triggers the displays clock. 010 - The Counter is triggered by counter #1 011 - The Counter is triggered by counter #2 100 - The Counter is triggered by counter #3 101 - The Counter is triggered by counter #4 110 - The Counter is triggered by counter #5 111 Counter is always on.

Table 42-258. DI1_SW_GEN1_9 Field Descriptions (continued)

Field	Description
24–16 di1_cnt_down_9	Counter falling edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's falling edge.
15 di1_tag_sel_9	Counter #9 can send a synchronous tag when counter #9 reach its predefined value or when it's triggering counter reaches its pre defined value. 1 - tag source is counter #9 0 - Tag's source is the triggering counter.
14-9	Reserved
8–0 di1_cnt_up_9	Counter rising edge position This parameter contains an integer part (bits 24:17) and a fractional part (bit 16). The position value is the amount of cycles between the trigger's start point and the waveform's rising edge.

42.2.3.10.22 DI1 Sync Assistance Gen Register (DI1_SYNC_AS_GEN)

Address 0xBASE+0xE048054 (DI1_SYNC_AS_GEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	di1_sync_start_en	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	di1_vsync_sel			0	di1_sync_start											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-258. DI0 Sync Assistance Gen Register (DI0_SYNC_AS_GEN)

Table 42-259. DI1_SYNC_AS_GEN Field Descriptions

Field	Description
31–29	Reserve
28 di1_sync_start_en	Reserved.
27–16	Reserved.

Table 42-259. DI1_SYNC_AS_GEN Field Descriptions (continued)

Field	Description
15–13 di1_vsync_sel	VSYNC select This field defines which of the counters functions as VSYNC signal 000 - VSYNC is coming from counter #1 001 - VSYNC is coming from counter #2 ... 111 - VSYNC is coming from counter #8
12	Reserved.
11–0 di1_sync_start	DI1 Sync start This field defines the number of low (including blanking rows) on the which the DI1 starts preparing the data for the next frame.

42.2.3.10.23 DI1 Data Wave Gen <i> Registers (DI1_DW_GEN_<i>)

The DI1_DW_GEN_<i> register holds pointers for the waveform generators.

These registers have different bit arrangements for parallel and serial display. When using a parallel display Table 42-260 is applicable. When using a serial interface Table 42-261 is applicable

Address 0xBASE+0xE048058 (DI1_DW_GEN_<i>)
offset 4
range 0 .. 11

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	di1_access_size_<i>								di1_componnent_size_<i>							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	di1_cst_<i>		di1_pt_6_<i>		di1_pt_5_<i>		di1_pt_4_<i>		di1_pt_3_<i>		di1_pt_2_<i>		di1_pt_1_<i>		di1_pt_0_<i>	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-259. DI0 Data Wave Gen <i> Registers (DI0_DW_GEN_<i>)

Table 42-260. DI1_DW_GEN <i> Field Descriptions

Field	Description
31–24 di1_access_size_<i>	DI1 Access Size <i> This field defines the amount of IPUv3EX cycles between any 2 accesses (an access may be a pixel or generic data that may have more one component)
23–16 di1_component_size_<i>	DI1 component Size This field defines the amount of IPUv3EX cycles between any 2 components
15-14 di1_cst_<i>	DI1 Chip Select pointer for waveform <i> This field points to a register that defines the waveform of the CS pin. The CS is automatically mapped to a specific display 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
13-12 di1_pt_6_<i>	DI1 PIN_17 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_17 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
11-10 di1_pt_5_<i>	DI1 PIN_16 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_16 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
9-8 di1_pt_4_<i>	DI1 PIN_15 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_15 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
7-6 di1_pt_3_<i>	DI1 PIN_14 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_14 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
5-4 di1_pt_2_<i>	DI1 PIN_13 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_13 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>

Table 42-260. DI1_DW_GEN <i> Field Descriptions (continued)

Field	Description
3-2 di1_pt_1_<i>	DI1 PIN_12 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_12 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
1-0 di1_pt_0_<i>	DI1 PIN_11 pointer for waveform <i> This field points to a register that defines the waveform of the PIN_11 pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>

Table 42-261. DI1_DW_GEN <i> Field Descriptions for serial display

Field	Description
31–24 di1_serial_period_<i>	DI1 Serial Period <i> This field defines the period of the time base serial display clock. The units are the internal DI clock
23–16 di1_start_period_<i>	DI1 start period This field defines the amount of cycles between the point where the access is ready to be launched to the actual point where the time base serial display clock restarts. The units are the internal DI clock
15-14 di1_cst_<i>	DI1 Chip Select pointer for waveform <i> This field points to a register that defines the waveform of the CS pin. For serial displays the down value as defined on DI1_DW_SET*_<i> is measured from the assertion of the last serial display time base clock. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
13-9	Reserved
8-4 di1_serial_valid_bits_<i>	DI1 Serial valid bits. This field defines the amount of valid bits to be transmitted within the 32 bits internal word aligned to bit[0]. The actual amount of valid bits is di1_serial_valid_bits_<i> + 1

Table 42-261. DI1_DW_GEN <i> Field Descriptions for serial display

Field	Description
3-2 di1_serial_rs_<i>	DI1 Serial RS This field points to a register that defines the waveform of the RS pin. For serial displays the down value as defined on DI1_DW_SET*_<i> is measured from the assertion of the last serial display time base clock. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>
1-0 di1_serial_clk_<i>	DI1 serial clock<i> This field points to a register that defines the waveform of the Serial clock pin. 00 - The waveform is defined according to the settings on DI1_DW_SET0_<i> 01- The waveform is defined according to the settings on DI1_DW_SET1_<i> 10 - The waveform is defined according to the settings on DI1_DW_SET2_<i> 11 - The waveform is defined according to the settings on DI1_DW_SET3_<i>

42.2.3.10.24 DI1 Data Wave Set 0 <i> Registers (DI1_DW_SET0_<i>)

Address 0xBASE+0xE048088 (DI1_DW_SET0_<i>) Access: User read/write
offset 4
range 0 .. 11

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	di1_data_cnt_down0_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	di1_data_cnt_up0_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-260. DI0 Data Wave Set 0 <i> Registers (DI0_DW_SET0_<i>)

Table 42-262. DI1_DW_SET0_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_data_cnt_down0_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>

Table 42-262. DI1_DW_SET0_<i>Field Descriptions (continued)

Field	Description
15–9	Reserved.
8–0 di1_data_cnt_up0_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>

42.2.3.10.25 DI1 Data Wave Set 1 <i> Registers (DI1_DW_SET1_<i>)

Address 0xBASE+0xE0480B8 (DI1_DW_SET1_<i>)
offset 4
range 0 .. 11

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								di1_data_cnt_down1_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								di1_data_cnt_up1_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-261. DI0 Data Wave Set 1 <i> Registers (DI0_DW_SET1_<i>)

Table 42-263. DI1_DW_SET1_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_data_cnt_down1_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>
15–9	Reserved.
8–0 di1_data_cnt_up1_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>

42.2.3.10.26 DI1 Data Wave Set 2 <i> Registers (DI1_DW_SET2_<i>)

Address 0xBASE+0xE0480E8 (DI1_DW_SET2_<i>) Access: User read/write
offset 4
range 0 .. 11

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								di1_data_cnt_down2_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								di1_data_cnt_up2_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-262. DI0 Data Wave Set 2 <i> Registers (DI0_DW_SET2_<i>)

Table 42-264. DI1_DW_SET2_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_data_cnt_down2_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>
15–9	Reserved.
8–0 di1_data_cnt_up2_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>

42.2.3.10.27 DI1 Data Wave Set 3 <i> Registers (DI1_DW_SET3_<i>)

Address 0xBASE+0xE048118 (DI1_DW_SET3_<i>) Access: User read/write
offset 4
range 0 .. 11

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								di1_data_cnt_down3_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								di1_data_cnt_up3_<i>								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-263. DI0 Data Wave Set 3 <i> Registers (DI0_DW_SET3_<i>)

Table 42-265. DI1_DW_SET3_<i>Field Descriptions

Field	Description
31–25	Reserved.
24–16 di1_data_cnt_down3_<i>	Waveform's falling edge position. This field defines the Waveform's falling edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>
15–9	Reserved.
8–0 di1_data_cnt_up3_<i>	Waveform's rising edge position. This field defines the Waveform's rising edge position. The Waveform is mapped to a pint according to the corresponding di1_pt_*_<i>

42.2.3.10.28 DI1 Step Repeat <i> Registers (DI1_STP_REP_<i>)

Address 0xBASE+0xE048148 (DI1_STP_REP_<i>)
offset 4
range 1 .. 4

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	di1_step_repeat_<2*i>											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					di1_step_repeat_<2*i-1>											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-264. DI0 Step Repeat <i> Registers (DI0_STP_REP_<i>)

Table 42-266. DI1_STP_REP_<i>Field Descriptions

Field	Description
27-16 11-0 di1_step_repeat_<i>	Step Repeat <i> This fields defines the amount of repetitions that will be performed by the counter <i>

42.2.3.10.29 DI1 Step Repeat 9 Registers (DI1_STP_REP_9)

Address 0xBASE+0xE048158 (DI1_STP_REP_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	di1_step_repeat_9											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-265. DI1 Step Repeat 9 Registers (DI1_STP_REP_9)

Table 42-267. DI1_STP_REP_9 Field Descriptions

Field	Description
27-16 11-0 di1_step_repeat_9	Step Repeat 9 This fields defines the amount of repetitions that will be performed by the counter 9

42.2.3.10.30 DI1 Serial Display Control Register (DI1_SER_CONF)

Address 0xBASE+0xE04815C (DI1_SER_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI1_SERIAL_LLA_PNTR_RS_R_1				DI1_SERIAL_LLA_PNTR_RS_R_0				DI1_SERIAL_LLA_PNTR_RS_W_1				DI1_SERIAL_LLA_PNTR_RS_W_0			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI1_SERIAL_LATCH								0	0	DI1_LLA_SER_ACCESS	DI1_SER_CLK_POLARITY	DI1_SERIAL_DATA_POLARITY	DI1_SERIAL_RS_POLARITY	DI1_SERIAL_CS_POLARITY	DI1_WAIT4SERIAL
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-266. DI0 Serial Display Control Register (DI0_SER_CONF)

Table 42-268. DI1_SER_CONF Field Descriptions

Field	Description
31–28 DI1_SERIAL_LLA_PNTR_RS_R_1	RS 3 waveform pointer for read low level access This pointer defines which waveform set will be chosen when the read low level access is targeted to RS group 1. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
27–24 DI1_SERIAL_LLA_PNTR_RS_R_0	RS 2 waveform pointer for low level access This pointer defines which waveform set will be chosen when the read low level access is targeted to RS group 0. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved

Table 42-268. DI1_SER_CONF Field Descriptions (continued)

Field	Description
23–20 DI1_SERIAL_LLA_PNTR_RS_W_1	RS 1 waveform pointer for write low level access This pointer defines which waveform set will be chosen when the write low level access is targeted to RS group 1. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
19–16 DI1_SERIAL_LLA_PNTR_RS_W_0	RS 0 waveform pointer for write low level access This pointer defines which waveform set will be chosen when the write low level access is targeted to RS group 0. 0000 Waveform set #1 0001Waveform set #2 ... 1011 Waveform set #12 1100 Reserved 1101 Reserved .... 1100 Reserved
15–8 DI1_SERIAL_LATCH	DI1 Serial Latch This field defines how many cycles to insert between serial read accesses start to data sampling in the IPUv3EX.
7–6	Reserved
5 DI1_LLA_SER_ACCESS	Direct Low Level Access to Serial display 1MCU access is performed via a direct path to the serial display in LLA mode, in this mode only the MCU in LLA mode can access the serial port 0 MCU access to the serial display port is not done directly, hence other source are allowed to access the serial port. The arbitration is done automatically
4 DI1_SER_CLK_POLARITY	Serial Clock Polarity The output polarity of the SER_CLK pin 1 - The clock is inverted 0 - The clock is not inverted
3 DI1_SERIAL_DATA_POLARITY	Serial Data Polarity The output polarity of the SER_DATA pin 1 - The data is inverted 0 - The data is not inverted
2 DI1_SERIAL_RS_POLARITY	Serial RS Polarity The output polarity of the SER_RS pin 1 - The RS is inverted 0 - The RS is not inverted

Table 42-268. DI1_SER_CONF Field Descriptions (continued)

Field	Description
1 DI1_SERIAL_CS_POLARITY	Serial Chip Select Polarity The output polarity of the SER_CS pin 1 - The CS is inverted 0 - The CS is not inverted
0 DI1_WAIT4SERIAL	Wait for Serial When the parallel display share pins with the serial port. Accessing the parallel port is not allowed till the serial port completes its access. 1 The parallel port should wait to the serial port as the pins are shared 0 The parallel port should not wait to the serial port as the pins are not shared

42.2.3.10.31 DI1 Special Signals Control Register (DI1_SSC)

Address 0xBASE+0xE048160 (DI1_SSC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	DI1_PIN17_ERM	DI1_PIN16_ERM	DI1_PIN15_ERM	DI1_PIN14_ERM	DI1_PIN13_ERM	DI1_PIN12_ERM	DI1_PIN11_ERM	DI1_CS_ERM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0		0	0	0	DI1_WAIT_ON	0	DI1_BYTE_EN_RD_IN	DI1_BYTE_EN_PNTR		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-267. DI0 Special Signals Control Register (DI0_SSC)

Table 42-269. DI1_SSC Field Descriptions

Field	Description
31–24	Reserved.
23 DI1_PIN17_ERM	DI1 PIN17 error recovery mode. This bit defines the error recovery mode of the PIN17 pin. 1 - The PIN17 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN17 pin following a display error detection

Table 42-269. DI1_SSC Field Descriptions (continued)

Field	Description
22 DI1_PIN16_ERM	DI1 PIN16 error recovery mode. This bit defines the error recovery mode of the PIN16 pin. 1 - The PIN16 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN16 pin following a display error detection
21 DI1_PIN15_ERM	DI1 PIN15 error recovery mode. This bit defines the error recovery mode of the PIN15 pin. 1 - The PIN15 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN15 pin following a display error detection
20 DI1_PIN14_ERM	DI1 PIN14 error recovery mode. This bit defines the error recovery mode of the PIN14 pin. 1 - The PIN14 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN14 pin following a display error detection
19 DI1_PIN13_ERM	DI1 PIN13 error recovery mode. This bit defines the error recovery mode of the PIN13 pin. 1 - The PIN13 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN13 pin following a display error detection
18 DI1_PIN12_ERM	DI1 PIN12 error recovery mode. This bit defines the error recovery mode of the PIN12 pin. 1 - The PIN12 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN12 pin following a display error detection
17 DI1_PIN11_ERM	DI1 PIN11 error recovery mode. This bit defines the error recovery mode of the PIN11 pin. 1 - The PIN11 pin is cleared in case of a synchronous display error. The clear will be done on the next VSYNC 0 - Nothing is done to the PIN11 pin following a display error detection
16 DI1_CS_ERM	DI1 GLUELOGIC error recovery mode. This bit defines the error recovery mode of the GLUELOGIC (see Table 42-425). 1 - The GLUELOGIC is release in case of a synchronous display error. The release will be done on the next VSYNC 0 - Nothing is done to the GLUELOGIC following a display error detection
15–6	Reserved.
5 DI1_WAIT_ON	Wait On This field defines the DC's response to WAIT signal 1 - The DC holds the flow as long as WAIT is asserted 0 - The DC continues the flow regardless the WAIT signal
4 DI1_BYTE_EN_POLARITY	Byte Enable polarity This bit defines the polarity of the byte enable signals to the display 1 - active high 0 active low

Table 42-269. DI1_SSC Field Descriptions (continued)

Field	Description
3 DI1_BYTE_EN_RD_IN	Byte Enable Read In This bit selects the source of the byte enable pins 1 The write byte enable signals are routed via bits [17:16] of the display's data, The read byte enable signals are routed via bits [19:18 of the display's data 0 The byte enable signals are routed via bits [17:16] of the display's data for both read and write
2-0 DI1_BYTE_EN_PNTR	Byte Enable Pointer This pointer selects the pin asserted along with the byte enables signals 000 wave form of byte enable as pin_11 001 wave form of byte enable as pin_12 ... 111 wave form of byte enable as suitable CS pin

42.2.3.10.32 DI1 Polarity Register (DI1_POL)

Address 0xBASE+0xE048164 (DI1_POL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	DI1_WAIT_POLARITY	DI1_CS1_BYTE_EN_POLARITY	DI1_CS0_BYTE_EN_POLARITY	DI1_CS1_DATA_POLARITY	di1_cs1_polarity_17	di1_cs1_polarity_16	di1_cs1_polarity_15	di1_cs1_polarity_14	di1_cs1_polarity_13	di1_cs1_polarity_12	di1_cs1_polarity_11
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI1_CS0_DATA_POLARITY	di1_cs0_polarity_17	di1_cs0_polarity_16	di1_cs0_polarity_15	di1_cs0_polarity_14	di1_cs0_polarity_13	di1_cs0_polarity_12	di1_cs0_polarity_11	DI1_DRDY_DATA_POLARITY	di1_drdy_polarity_17	di1_drdy_polarity_16	di1_drdy_polarity_15	di1_drdy_polarity_14	di1_drdy_polarity_13	di1_drdy_polarity_12	di1_drdy_polarity_11
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-268. DI1 Polarity Register (DI1_POL)

Table 42-270. DI1_POL Field Descriptions

Field	Description
31-27	Reserved.
26 DI1_WAIT_POLARITY	WAIT polarity This bit defines the polarity of the wait signal input coming from the displa1 1 - active high 0 active low
25 DI1_CS1_BYTE_EN_POLARITY	Byte Enable associated with CS1 polarity This bit defines the polarity of the byte enable signals to the display 1 - active high 0 active low
24 DI1_CS0_BYTE_EN_POLARITY	Byte Enable associated with CS0 polarity This bit defines the polarity of the byte enable signals to the display 1 - active high 0 active low
23 DI1_CS1_DATA_POLARITY	Data Polarity associated with CS1
22-16 di1_cs1_polarity_[17:11]	DI1 output pin's polarity for CS1 This bits define the polarity of each of the DI's outputs when CS1 is asserted 1 The output pin is active high 0 The output pin is active low
15 DI1_CS0_DATA_POLARITY	Data Polarity associated with CS0
14-8 di1_cs0_polarity_[17:11]	DI1 output pin's polarity for CS1 This bits define the polarity of each of the DI's outputs when CS1 is asserted 1 The output pin is active high 0 The output pin is active low
7 DI1_DRDY_DATA_POLARITY	Data Polarity associated with DRDY
6-0 di1_drdy_polarity_[17:11]	DI1 output dynamic pin's polarity for synchronous access This bits define the polarity of each of the DI's outputs when synchronous display access is asserted 1 The output pin is active high 0 The output pin is active low The pins' default polarity is the same as defined in the di0_drdy_polarity_[17:11] bits

42.2.3.10.33 DI1 Active Window 0 Register (DI1_AW0)

Address 0xBASE+0xE048168 (DI1_AW0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI1_AW_TRIG_SEL				DI1_AW_HEND											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI1_AW_HCOUNT_SEL				DI1_AW_HSTART											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-269. DI0 Active Window 0 Register (DI0_AW0)

Table 42-271. DI1_AW0 Field Descriptions

Field	Description
31-28 DI1_AW_TRIG_SEL	This field selects the trigger for sending data during the display's active window 000 - disabled 001 - The trigger is the same trigger that triggers the displays clock. 010 - The trigger is counter #1 011 - The trigger is counter #2 100 - The trigger is counter #3 101 - The trigger is counter #4 110 - The trigger is counter #5 111 The trigger is always on.
27-16 DI1_AW_HEND	This field defines the horizontal end of the active window
15-12 DI1_AW_HCOUNT_SEL	This field selects the counter that counts the horizontal position of the display's active window 0000 - disabled 0001 - reserved 0010 - The counter is counter #1 0011 - The counter is counter #2 0100 - The counter is counter #3 0101 - The counter is counter #4 0110 - The counter is counter #5 ..... 1001 - The counter is counter #8
11-0 DI1_AW_HSTART	This field defines the horizontal start of the active window DI1_AW_HSTART < DI1_AW_HEND

42.2.3.10.34 DI1 Active Window 1 Register (DI1_AW1)

Address 0xBASE+0xE04816C (DI1_AW1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	DI1_AW_VEND											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI1_AW_VCOUNT_SEL				DI1_AW_VSTART											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-270. DI0 Active Window 1 Register (DI0_AW1)

Table 42-272. DI1_AW1 Field Descriptions

Field	Description
31-28	Reserved
27-16 DI1_AW_VEND	This field defines the vertical end of the active window
15-12 DI1_AW_VCOUNT_SEL	This field selects the counter that counts the vertical position of the display's active window 0000 - disabled 0001 - reserved 0010 - The counter is counter #1 0011 - The counter is counter #2 0100 - The counter is counter #3 0101 - The counter is counter #4 0110 - The counter is counter #5 ..... 1001 - The counter is counter #8
11-0 DI1_AW_VSTART	This field defines the vertical start of the active window DI1_AW_VSTART < DI1_AW_VEND

42.2.3.10.35 DI1 Screen Configuration Register (DI1_SCR_CONF)

Address 0xBASE+0xE048170 (DI1_SCR_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	DI1_SCREEN_HEIGHT											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-271. DI1 Screen Configuration Register (DI1_SCR_CONF)

Table 42-273. DI1_SCR_CONF Field Descriptions

Field	Description
31–12	Reserved.
11–0 DI1_SCREEN_HEIGHT	This field defines the number of display rows (Number_of_ROWS = DI1_SCREEN_HEIGHT+1) This field is used for VSYNC calculation and for anti-tearing

42.2.3.10.36 DI1 Status Register (DI1_STAT)

Address 0xBASE+0xE048174 (DI1_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	DI1_CNTR_FIFO_FULL	DI1_CNTR_FIFO_EMPTY	DI1_READ_FIFO_FULL	DI1_READ_FIFO_EMPTY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Figure 42-272. DI1 Status Register (DI1_STAT)

Table 42-274. DI1_STAT Field Descriptions

Field	Description
31-4	Reserved
3 DI1_CNTR_FIFO_FULL	
2 DI1_READ_CNTR_EMPTY	
1 DI1_READ_FIFO_FULL	
0 DI1_READ_FIFO_EMPTY	

42.2.3.11 SMFC Registers

42.2.3.11.1 SMFC Mapping Register (SMFC_MAP)

The purpose of this register is to map CSI frames to IDMAC channels.

Address 0xBASE+0xE050000 (SMFC_MAP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	MAP_CH3				MAP_CH2				MAP_CH1			MAP_CH0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-273. SMFC Mapping Register (SMFC_MAP)

Table 42-275. SMFC_MAP Field Descriptions

Field	Description
31–12	Reserved.
11-9 MAP_CH3	DMASMFC channel 3mapping bits. 000 CSI0, ID=0 mapped to DMASMFC channel 3. 001 CSI0, ID=1 mapped to DMASMFC channel 3. 010 CSI0, ID=2 mapped to DMASMFC channel 3. 011 CSI0, ID=3 mapped to DMASMFC channel 3. 100 CSI1, ID=0 mapped to DMASMFC channel 3. 101 CSI1, ID=1 mapped to DMASMFC channel 3. 110 CSI1, ID=2 mapped to DMASMFC channel 3. 111 CSI1, ID=3 mapped to DMASMFC channel 3.
8-6 MAP_CH2	DMASMFC channel 2 mapping bits. 000 CSI0, ID=0 mapped to DMASMFC channel 2. 001 CSI0, ID=1 mapped to DMASMFC channel 2. 010 CSI0, ID=2 mapped to DMASMFC channel 2. 011 CSI0, ID=3 mapped to DMASMFC channel 2. 100 CSI1, ID=0 mapped to DMASMFC channel 2. 101 CSI1, ID=1 mapped to DMASMFC channel 2. 110 CSI1, ID=2 mapped to DMASMFC channel 2. 111 CSI1, ID=3 mapped to DMASMFC channel 2.

Table 42-275. SMFC_MAP Field Descriptions (continued)

Field	Description
5-3 MAP_CH1	DMASMFC channel 1 mapping bits. 000 CSI0, ID=0 mapped to DMASMFC channel 1. 001 CSI0, ID=1 mapped to DMASMFC channel 1. 010 CSI0, ID=2 mapped to DMASMFC channel 1. 011 CSI0, ID=3 mapped to DMASMFC channel 1. 100 CSI1, ID=0 mapped to DMASMFC channel 1. 101 CSI1, ID=1 mapped to DMASMFC channel 1. 110 CSI1, ID=2 mapped to DMASMFC channel 1. 111 CSI1, ID=3 mapped to DMASMFC channel 1.
2-0 MAP_CH0	DMASMFC channel 0 mapping bits. 000 CSI0, ID=0 mapped to DMASMFC channel 0. 001 CSI0, ID=1 mapped to DMASMFC channel 0. 010 CSI0, ID=2 mapped to DMASMFC channel 0. 011 CSI0, ID=3 mapped to DMASMFC channel 0. 100 CSI1, ID=0 mapped to DMASMFC channel 0. 101 CSI1, ID=1 mapped to DMASMFC channel 0. 110 CSI1, ID=2 mapped to DMASMFC channel 0. 111 CSI1, ID=3 mapped to DMASMFC channel 0.

42.2.3.11.2 SMFC Water Mark Control Register (SMFC_WMC)

The purpose of this register is to control watermarks levels of DMA channels. The bit setting given relative to FIFO size and not in number of words since FIFO size depend from number of enabled DMA channels.

Address 0xBASE+0xE050004 (SMFC_WMC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	WM3_CLR				WM3_SET				WM2_CLR			
W														WM2_SET		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	WM1_CLR				WM1_SET				WM0_CLR			
W														WM0_SET		
Reset	0	0	0	0	1	0	0	1	1	0	1	0	0	1	1	0

Figure 42-274. SMFC Water Mark Control Register (SMFC_WMC)

Table 42-276. SMFC_MAP Field Descriptions

Field	Description
31–28	Reserved.
27-25 WM3_CLR	Watermark “clear” level of DMASMFC channel 3. 000 clear watermark level when FIFO is full on 1/8 of their size. 001 clear watermark level when FIFO is full on 2/8 of their size. ... 110clear watermark level when FIFO is full on 7/8 of their size. 111clear watermark level when FIFO is full.
24-22 WM3_SET	Watermark “set” level of DMASMFC channel 3 000 set watermark level when FIFO is full on 1/8 of their size. 001 set watermark level when FIFO is full on 2/8 of their size. ... 110set watermark level when FIFO is full on 7/8 of their size. 111set watermark level when FIFO is full.
21-19 WM2_CLR	Watermark “clear” level of DMASMFC channel 2. 000 clear watermark level when FIFO is full on 1/8 of their size. 001 clear watermark level when FIFO is full on 2/8 of their size. ... 110clear watermark level when FIFO is full on 7/8 of their size. 111clear watermark level when FIFO is full.
18-16 WM2_SET	Watermark “set” level of DMASMFC channel 2. 000 set watermark level when FIFO is full on 1/8 of their size. 001 set watermark level when FIFO is full on 2/8 of their size. ... 110set watermark level when FIFO is full on 7/8 of their size. 111set watermark level when FIFO is full.
15-12	Reserved.
11-9 WM1_CLR	Watermark “clear” level of DMASMFC channel 1. 000 clear watermark level when FIFO is full on 1/8 of their size. 001 clear watermark level when FIFO is full on 2/8 of their size. ... 110clear watermark level when FIFO is full on 7/8 of their size. 111clear watermark level when FIFO is full.
8-6 WM1_SET	Watermark “set” level of DMASMFC channel 1. 000 set watermark level when FIFO is full on 1/8 of their size. 001 set watermark level when FIFO is full on 2/8 of their size. ... 110set watermark level when FIFO is full on 7/8 of their size. 111set watermark level when FIFO is full.

Table 42-276. SMFC_MAP Field Descriptions (continued)

Field	Description
5-3 WM0_CLR	Watermark “clear” level of DMASMFC channel 0. 000 clear watermark level when FIFO is full on 1/8 of their size. 001 clear watermark level when FIFO is full on 2/8 of their size. ... 110clear watermark level when FIFO is full on 7/8 of their size. 111clear watermark level when FIFO is full.
2-0 WM0_SET	Watermark “set” level of DMASMFC channel 0. 000 set watermark level when FIFO is full on 1/8 of their size. 001 set watermark level when FIFO is full on 2/8 of their size. ... 110set watermark level when FIFO is full on 7/8 of their size. 111set watermark level when FIFO is full.

42.2.3.11.3 SMFC Burst Size Register (SMFC_BS)

This register holds the burst size value for each DMASMFC channel. The burst size is the number of IDMAC’s active accesses that will done for each IDMAC’s burst. This number is a function of PFS, BPP & NPB parameters in the IDMAC’s CPMEM. These are the parameters corresponding to the IDMAC’s channel used. The table below describes what should be the burst size according to PFS, BPP & NPB settings

Table 42-277. SMFC Burst Size

BPP	PFS	BURST_SIZE
8	6	NPB[6:4]
16	6	NPB[6:4]
All other	All other	NPB[6:2]

Address 0xBASE+0xE050008 (SMFC_BS) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BURST3_SIZE				BURST2_SIZE				BURST1_SIZE				BURST0_SIZE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-275. SMFC Burst Size Register (SMFC_BS)

Table 42-278. SMFC_BS Field Descriptions

Field	Description
31–16	Reserved.
3-0 BURST3 _SIZE	Burst Size of SMFCDMA channel 3. The value programed here should be BURST_SIZE-1 (for example: for a burst size of 1, write 0 to these bits)
3-0 BURST2 _SIZE	Burst Size of SMFCDMA channel 2. The value programed here should be BURST_SIZE-1 (for example: for a burst size of 1, write 0 to these bits)
3-0 BURST1 _SIZE	Burst Size of SMFCDMA channel 1. The value programed here should be BURST_SIZE-1 (for example: for a burst size of 1, write 0 to these bits)
3-0 BURST0 _SIZE	Burst Size of SMFCDMA channel 0. The value programed here should be BURST_SIZE-1 (for example: for a burst size of 1, write 0 to these bits)

42.2.3.12 DC Registers

42.2.3.12.1 DC Read Channel Configuration Register (DC_READ_CH_CONF)

Address 0xBASE+0xE058000 (DC_READ_CH_CONF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TIME_OUT_VALUE															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	CS_ID_3	CS_ID_2	CS_ID_1	CS_ID_0	0	CHAN_MASK_DEFAULT_0	W_SIZE_0		PROG_DISP_ID_0		PROG_DI_ID_0	RD_CHANNEL_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-276. DC Read Channel Configuration Register (DC_READ_CH_CONF)

Table 42-279. DC_READ_CH_CONF Field Descriptions

Field	Description
31–16 TIME_OUT_VALUE	Time out value. In case of a error during read accesses to the display, where no response from the display was received. A time-out counter will terminate the current access and perform the next commend defined in the microcode. This field defines the amount of the hsp_clk cycles counted before the time-out event is issued. This event is tied to the interrupt controller and can generate an error interrupt.
11 CS_ID_3	This bit maps an asynchronous display to a chip select 1 - display #3 is connected to CS1 0 - display #3 is connected to CS0
10 CS_ID_2	This bit maps an asynchronous display to a chip select 1 - display #2 is connected to CS1 0 - display #2 is connected to CS0
9 CS_ID_1	This bit maps an asynchronous display to a chip select 1 - display #1 is connected to CS1 0 - display #1 is connected to CS0
8 CS_ID_0	This bit maps an asynchronous display to a chip select 1 - display #0 is connected to CS1 0 - display #0 is connected to CS0
7	Reserved
6 CHAN_MASK_DEFAULT_0	Event mask bit for the read channel When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
5–4 W_SIZE_0	Word Size The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used
3–2 PROG_DISP_ID_0	The field defines which one of the 4 displays can be read. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
1 PROG_DI_ID_0	This bit select the DI which a read transaction can be performed through 1 - DI #1 0 - DI #0
0 RD_CHANNEL_EN	This bit enables the read channel. 1 - The Read channel is enabled 0 - The Read channel is disabled

42.2.3.12.2 DC Read Channel Start Address Register (DC_READ_CH_ADDR)

Address 0xBASE+0xE058004 (DC_READ_CH_ADDR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-277. DC Read Channel Start Address Register (DC_READ_CH_ADDR)

Table 42-280. DC_READ_CH_ADDR Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_0	This field defines the start address within the display's memory space where the read transactions will be done from.

42.2.3.12.3 DC Routine Link Register 0 Channel 0 (DC_RL0_CH_0)

Address 0xBASE+0xE058008 (DC_RL0_CH_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NL_START_CHAN_0								0	0	0	0	COD_NL_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NF_START_CHAN_0								0	0	0	0	COD_NF_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-278. DC Routine Link Register 0 Channel 0 (DC_RL0_CH_0)

Table 42-281. DC_RL0_CH_0 Field Descriptions

Field	Description
31–24 COD_NL_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new line event (NL) resides
23–20	Reserved.
19–16 COD_NL_PRIORITY_CHAN_0	This field defines the priority of the new line (NL) event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NF_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new Frame event (NF) resides
7–4	Reserved.
3–0 COD_NF_PRIORITY_CHAN_0	This field defines the priority of the new frame (NF) event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.4 DC Routine Link Register 1 Channel 0 (DC_RL1_CH_0)

Address 0xBASE+0xE05800C (DC_RL1_CH_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NFIELD_START_CHAN_0								0	0	0	0	COD_NFIELD_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOF_START_CHAN_0								0	0	0	0	COD_EOF_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-279. DC Routine Link Register 1 Channel 0 (DC_RL1_CH_0)

Table 42-282. DC_RL1_CH_0 Field Descriptions

Field	Description
31–24 COD_NFIELD_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new field event resides
23–20	Reserved.
19–16 COD_NFIELD_PRIORITY_CHAN_0	This field defines the priority of the new field event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOF_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the end-of-frame event (EOF) resides
7–4	Reserved.
3–0 COD_EOF_PRIORITY_CHAN_0	This field defines the priority of the end-of-frame event (EOF) event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.5 DC Routine Link Register 2 Channel 0 (DC_RL2_CH_0)

Address 0xBASE+0xE058010 (DC_RL2_CH_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_EOFIELD_START_CHAN_0								0	0	0	0	COD_EOFIELD_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOL_START_CHAN_0								0	0	0	0	COD_EOL_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-280. DC Routine Link Register 2 Channel 0 (DC_RL2_CH_0)

Table 42-283. DC_RL2_CH_0 Field Descriptions

Field	Description
31–24 COD_EOFIELD_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the end-of-field event resides
23–20	Reserved.
19–16 COD_EOFIELD_PRIORITY_CHAN_0	This field defines the priority of the end-of-field event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOL_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the end-of-line event (EOL) resides

Table 42-283. DC_RL2_CH_0 Field Descriptions (continued)

Field	Description
7–4	Reserved.
3–0 COD_EOL_PRIORITY_CHAN_0	This field defines the priority of the end-of-line event (EOL) event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.6 DC Routine Link Register 3 Channel 0 (DC_RL3_CH_0)

Address 0xBASE+0xE058014 (DC_RL3_CH_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_0								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_0								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-281. DC Routine Link Register 3 Channel 0 (DC_RL3_CH_0)

Table 42-284. DC_RL3_CH_0 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides
23–20	Reserved.

Table 42-284. DC_RL3_CH_0 Field Descriptions (continued)

Field	Description
19–16 COD_NEW_CHAN_PRIORITY_CHAN_0	This field defines the priority of the new channel event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NEW_ADDR_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_0	This field defines the priority of the new address event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.7 DC Routine Link Register 4 Channel 0 (DC_RL4_CH_0)

Address 0xBASE+0xE058018 (DC_RL4_CH_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_0								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-282. DC Routine Link Register 4 Channel 0 (DC_RL4_CH_0)

Table 42-285. DC_RL4_CH_0 Field Descriptions

Field	Description
31–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_0	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_0	This field defines the priority of the new data event 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.8 DC Write Channel 1 Configuration Register (DC_WR_CH_CONF_1)

Address 0xBASE+0xE05801C (DC_WR_CH_CONF_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	PROG_START_TIME_1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	FIELD_MODE_1	CHAN_MASK_DEFAULT_1	PROG_CHAN_TYP_1			PROG_DISP_ID_1		PROG_DI_ID_1	W_SIZE_1	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-283. DC Write Channel 1 Configuration Register (DC_WR_CH_CONF_1)

Table 42-286. DC_WR_CH_CONF_1 Field Descriptions

Field	Description
31–27	Reserved.
26–16 PROG_ START_ TIME_1	This field defines the delay between display ‘s vertical synchronization pulse and the start time point of DC’s channel 1 window. The delay is defined in pairs of rows. It is used for tearing elimination
15–9	Reserved.
9 FIELD_ MODE_1	Field mode bit for channel #1 This bit defines if the channel works in field mode or frame mode; This bit is relevant if the flow is sync flow 1 - Field mode 0 - Frame mode
8 CHAN_ MASK_D EFAULT _1	Event mask bit for channel #1 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
7–5 PROG_ CHAN_T YP_1	This field define the mode of operation of channel #1 000 - Disable 001 - Reserved 010 - Reserved 100 - Normal mode without anti-tearing. For sync display this is the only mode allowed 101 - Normal mode with anti-tearing 110 - Reserved 111 - Additional command channel is added to the flow handled by DC channel #1
4–3 PROG_ DISP_ID _1	The field defines which one of the 4 displays is associated with channel #1. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 PROG_ DI_ID_1	This bit select the DI which a transaction associated with channel #1 can be performed to 1 - DI #1 0 - DI #0
1-0 W_SIZE _1	Word Size associated with channel #1 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.9 DC Write Channel 1 Configuration Register (DC_WR_CH_ADDR_1)

Address 0xBASE+0xE058020 (DC_WR_CH_ADDR_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-284. DC Write Channel 1 Configuration Register (DC_WR_CH_ADDR_1)

Table 42-287. DC_WR_CH_ADDR_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_1	This field defines the start address within the display's memory space where the write transactions will be done to for channel #1.

42.2.3.12.10 DC Routine Link Register 0 Channel 1 (DC_RL0_CH_1)

Address 0xBASE+0xE058024 (DC_RL0_CH_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NL_START_CHAN_1								0	0	0	0	COD_NL_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NF_START_CHAN_1								0	0	0	0	COD_NF_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-285. DC Routine Link Register 0 Channel 1 (DC_RL0_CH_1)

Table 42-288. DC_RL0_CH_1 Field Descriptions

Field	Description
31–24 COD_NL_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new line event resides (associated with channel #1)
23–20	Reserved.
19–16 COD_NL_PRIORITY_CHAN_1	This field defines the priority of the new line event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NF_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new frame event resides (associated with channel #1)
7–4	Reserved.
3–0 COD_NF_PRIORITY_CHAN_1	This field defines the priority of the new frame event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.11 DC Routine Link Register 1 Channel 1 (DC_RL1_CH_1)

Address 0xBASE+0xE058028 (DC_RL1_CH_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NFIELD_START_CHAN_1								0	0	0	0	COD_NFIELD_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOF_START_CHAN_1								0	0	0	0	COD_EOF_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-286. DC Routine Link Register 1 Channel 1 (DC_RL1_CH_1)

Table 42-289. DC_RL1_CH_1 Field Descriptions

Field	Description
31–24 COD_NFIELD_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new field event resides (associated with channel #1)
23–20	Reserved.
19–16 COD_NFIELD_PRIORITY_CHAN_1	This field defines the priority of the new field event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOF_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the end of frame event resides (associated with channel #1)
7–4	Reserved.
3–0 COD_EOF_PRIORITY_CHAN_1	This field defines the priority of the end of frame event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.12 DC Routine Link Register 2 Channel 1 (DC_RL2_CH_1)

Address 0xBASE+0xE05802C (DC_RL2_CH_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_EOFIELD_START_CHAN_1								0	0	0	0	COD_EOFIELD_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOL_START_CHAN_1								0	0	0	0	COD_EOL_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-287. DC Routine Link Register 2 Channel 1 (DC_RL2_CH_1)

Table 42-290. DC_RL2_CH_1 Field Descriptions

Field	Description
31–24 COD_EOFIELD_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the end of field event resides (associated with channel #1)
23–20	Reserved.
19–16 COD_EOFIELD_PRIORITY_CHAN_1	This field defines the priority of the end of field event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOL_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the end of line event resides (associated with channel #1)

Table 42-290. DC_RL2_CH_1 Field Descriptions (continued)

Field	Description
7–4	Reserved.
3–0 COD_EOL_PRIORITY_CHAN_1	This field defines the priority of the end of line event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.13 DC Routine Link Register 3 Channel 1 (DC_RL3_CH_1)

Address 0xBASE+0xE058030 (DC_RL3_CH_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_1								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_1								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-288. DC Routine Link Register 3 Channel 1 (DC_RL3_CH_1)

Table 42-291. DC_RL3_CH_1 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #1)
23–20	Reserved.

Table 42-291. DC_RL3_CH_1 Field Descriptions (continued)

Field	Description
19–16 COD_NEW_CHAN_PRIORITY_CHAN_1	This field defines the priority of the new channel event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NEW_ADDR_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #1)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_1	This field defines the priority of the new address event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.14 DC Routine Link Register 4 Channel 1 (DC_RL4_CH_1)

Address 0xBASE+0xE058034 (DC_RL4_CH_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_1								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_1			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-289. DC Routine Link Register 4 Channel 1 (DC_RL4_CH_1)

Table 42-292. DC_RL4_CH_1 Field Descriptions

Field	Description
31–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_1	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #1)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_1	This field defines the priority of the new data event (associated with channel #1) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.15 DC Write Channel 2 Configuration Register (DC_WR_CH_CONF_2)

Address 0xBASE+0xE058038 (DC_WR_CH_CONF_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	PROG_START_TIME_2										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CHAN_MASK_DEFAULT_2	PROG_CHAN_TYP_2			PROG_DISP_ID_2		PROG_DI_ID_2	W_SIZE_2	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-290. DC Write Channel 2 Configuration Register (DC_WR_CH_CONF_2)

Table 42-293. DC_WR_CH_CONF_2 Field Descriptions

Field	Description
31–27	Reserved.
26–16 PROG_ START_ TIME_2	This field defines the delay between display ‘s vertical synchronization pulse and the start time point of DC’s channel 2 window. The delay is defined in pairs of rows. It is used for tearing elimination
15–9	Reserved.
8 CHAN_ MASK_D EFAULT _2	Event mask bit for channel #2 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
7–5 PROG_ CHAN_T YP_2	This field define the mode of operation of channel #2 000 - Disable 001 - Reserved 010 - Reserved 100 - Normal mode without anti-tearing. For sync display this is the only mode allowed 101 - Normal mode with anti-tearing 110 - Reserved 111 - Additional command channel is added to the flow handled by DC channel #2
4–3 PROG_ DISP_ID _2	The field defines which one of the 4 displays is associated with channel #2. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 PROG_ DI_ID_2	This bit select the DI which a transaction associated with channel #2 can be performed to 1 - DI #1 0 - DI #0
1-0 W_SIZE _2	Word Size The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.16 DC Write Channel 2 Configuration Register (DC_WR_CH_ADDR_2)

Address 0xBASE+0xE05803C (DC_WR_CH_ADDR_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_2												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-291. DC Write Channel 2 Configuration Register (DC_WR_CH_ADDR_2)

Table 42-294. DC_WR_CH_ADDR_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_2	This field defines the start address within the display's memory space where the write transactions will be done to for channel #2.

42.2.3.12.17 DC Routine Link Register 0 Channel 2 (DC_RL0_CH_2)

Address 0xBASE+0xE058040 (DC_RL0_CH_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NL_START_CHAN_2								0	0	0	0	COD_NL_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NF_START_CHAN_2								0	0	0	0	COD_NF_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-292. DC Routine Link Register 0 Channel 2 (DC_RL0_CH_2)

Table 42-295. DC_RL0_CH_2 Field Descriptions

Field	Description
31–24 COD_NL_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new line event resides (associated with channel #2)
23–20	Reserved.
19–16 COD_NL_PRIORITY_CHAN_2	This field defines the priority of the new line event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NF_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new frame event resides (associated with channel #2)
7–4	Reserved.
3–0 COD_NF_PRIORITY_CHAN_2	This field defines the priority of the new frame event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.18 DC Routine Link Register 1 Channel 2 (DC_RL1_CH_2)

Address 0xBASE+0xE058044 (DC_RL1_CH_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NFIELD_START_CHAN_2								0	0	0	0	COD_NFIELD_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOF_START_CHAN_2								0	0	0	0	COD_EOF_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-293. DC Routine Link Register 1 Channel 2 (DC_RL1_CH_2)

Table 42-296. DC_RL1_CH_2 Field Descriptions

Field	Description
31–24 COD_NFIELD_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new field event resides (associated with channel #2)
23–20	Reserved.
19–16 COD_NFIELD_PRIORITY_CHAN_2	This field defines the priority of the new field event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOF_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the end of frame event resides (associated with channel #2)
7–4	Reserved.
3–0 COD_EOF_PRIORITY_CHAN_2	This field defines the priority of the end of frame event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.19 DC Routine Link Register 2 Channel 2 (DC_RL2_CH_2)

Address 0xBASE+0xE058048 (DC_RL2_CH_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_EOFIELD_START_CHAN_2								0	0	0	0	COD_EOFIELD_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOL_START_CHAN_2								0	0	0	0	COD_EOL_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-294. DC Routine Link Register 2 Channel 2 (DC_RL2_CH_2)

Table 42-297. DC_RL2_CH_2 Field Descriptions

Field	Description
31–24 COD_EOFIELD_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the end of field event resides (associated with channel #2)
23–20	Reserved.
19–16 COD_EOFIELD_PRIORITY_CHAN_2	This field defines the priority of the end of field event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOL_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the end of line event resides (associated with channel #2)

Table 42-297. DC_RL2_CH_2 Field Descriptions (continued)

Field	Description
7–4	Reserved.
3–0 COD_EOL_PRIORITY_CHAN_2	This field defines the priority of the end of line event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.20 DC Routine Link Register 3 Channel 2 (DC_RL3_CH_2)

Address 0xBASE+0xE05804C (DC_RL3_CH_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_2								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_2								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-295. DC Routine Link Register 3 Channel 2 (DC_RL3_CH_2)

Table 42-298. DC_RL3_CH_2 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #2)
23–20	Reserved.

Table 42-298. DC_RL3_CH_2 Field Descriptions (continued)

Field	Description
19–16 COD_NEW_CHAN_PRIORITY_CHAN_2	This field defines the priority of the new channel event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NEW_ADDR_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #2)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_2	This field defines the priority of the new address event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.21 DC Routine Link Register 4 Channel 2 (DC_RL4_CH_2)

Address 0xBASE+0xE058050 (DC_RL4_CH_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_2								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-296. DC Routine Link Register 4 Channel 2 (DC_RL4_CH_2)

Table 42-299. DC_RL4_CH_2 Field Descriptions

Field	Description
31–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_2	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #2)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_2	This field defines the priority of the new data event (associated with channel #2) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.22 DC Command Channel 3 Configuration Register (DC_CMD_CH_CONF_3)

Address 0xBASE+0xE058054 (DC_CMD_CH_CONF_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_CMND_START_CHAN_RS1_3								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_CMND_START_CHAN_RS0_3								0	0	0	0	0	0	W_SIZE_3	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-297. DC Command Channel 3 Configuration Register (DC_CMD_CH_CONF_3)

Table 42-300. DC_CMD_CH_CONF_3 Field Descriptions

Field	Description
31–24 COD_CMND_START_CHAN_RS1_3	This field is a pointer to the address within the microcode memory where the routine that handles the command start event resides (associated with channel #3); This field is relevant when RS is equal to 1
23–16	Reserved.

Table 42-300. DC_CMD_CH_CONF_3 Field Descriptions (continued)

Field	Description
15–8 COD_CMND_START_CHAN_RS0_3	This field is a pointer to the address within the microcode memory where the routine that handles the command start event resides (associated with channel #3); This field is relevant when RS is equal to 0
7–2	Reserved.
1–0 W_SIZE_3	Word Size associated with channel #3 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.23 DC Command Channel 4 Configuration Register (DC_CMD_CH_CONF_4)

Address 0xBASE+0xE058058 (DC_CMD_CH_CONF_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_CMND_START_CHAN_RS1_4								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_CMND_START_CHAN_RS0_4								0	0	0	0	0	0	W_SIZE_4	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-298. DC Command Channel 4 Configuration Register (DC_CMD_CH_CONF_4)

Table 42-301. DC_CMD_CH_4 Field Descriptions

Field	Description
31–24 COD_CMND_START_CHAN_RS1_4	This field is a pointer to the address within the microcode memory where the routine that handles the command start event resides (associated with channel #4); This field is relevant when RS is equal to 1
23–16	Reserved.

Table 42-301. DC_CMD_CH_4 Field Descriptions (continued)

Field	Description
15–8 COD_CMND_START_CHAN_RS0_4	This field is a pointer to the address within the microcode memory where the routine that handles the command start event resides (associated with channel #4); This field is relevant when RS is equal to 0
7–2	Reserved.
1–0 W_SIZE_4	Word Size associated with channel #4 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.24 DC Write Channel 5 Configuration Register (DC_WR_CH_CONF_5)

Address 0xBASE+0xE05805C (DC_WR_CH_CONF_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	PROG_START_TIME_5										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	FIELD_MODE_5	CHAN_MASK_DEFAULT_5	PROG_CHAN_TYP_5			PROG_DISP_ID_5		PROG_DI_ID_5	W_SIZE_5	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-299. DC Write Channel 5 Configuration Register (DC_WR_CH_CONF_5)

Table 42-302. DC_WR_CH_CONF_5 Field Descriptions

Field	Description
31–27	Reserved.
26–16 PROG_START_TIME_5	This field defines the delay between display ‘s vertical synchronization pulse and the start time point of DC’s channel 5 window. The delay is defined in pairs of rows. It is used for tearing elimination
15–9	Reserved.

Table 42-302. DC_WR_CH_CONF_5 Field Descriptions (continued)

Field	Description
9 FIELD_ MODE_5	Field mode bit for channel #5 This bit defines if the channel works in field mode or frame mode; This bit is relevant if the flow is sync flow 1 - Field mode 0 - Frame mode
8 CHAN_ MASK_D EFAULT _5	Event mask bit for channel #5 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
7–5 PROG_ CHAN_T YP_5	This field define the mode of operation of channel #5 000 - Disable 001 - Reserved 010 - Reserved 100 - Normal mode without anti-tearing. For sync display this is the only mode allowed 101 - Normal mode with anti-tearing 110 - Reserved 111 - Additional command channel is added to the flow handled by DC channel #5
4–3 PROG_ DISP_ID _5	The field defines which one of the 4 displays is associated with channel #5. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 PROG_ DI_ID_5	This bit select the DI which a transaction associated with channel #5 can be performed to. When channel 28 is connected to DI0, channel 23 must be connected to DI1 even if ch23 is not used. This is done by writing 1 to this bit. 1 - DI #1 0 - DI #0
1-0 W_SIZE _5	Word Size associated with channel #5 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.25 DC Write Channel 5 Configuration Register (DC_WR_CH_ADDR_5)

Address 0xBASE+0xE058060 (DC_WR_CH_ADDR_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_5												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_5															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-300. DC Write Channel 5 Configuration Register (DC_WR_CH_ADDR_5)

Table 42-303. DC_WR_CH_ADDR_5 Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_5	This field defines the start address within the display's memory space where the write transactions will be done to for channel #5.

42.2.3.12.26 DC Routine Link Register 0 Channel 5 (DC_RL0_CH_5)

Address 0xBASE+0xE058064 (DC_RL0_CH_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NL_START_CHAN_5								0	0	0	0	COD_NL_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NF_START_CHAN_5								0	0	0	0	COD_NF_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-301. DC Routine Link Register 0 Channel 5 (DC_RL0_CH_5)

Table 42-304. DC_RL0_CH_5 Field Descriptions

Field	Description
31–24 COD_NL_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new line event resides (associated with channel #5)
23–20	Reserved.
19–16 COD_NL_PRIORITY_CHAN_5	This field defines the priority of the new line event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NF_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new frame event resides (associated with channel #5)
7–4	Reserved.
3–0 COD_NF_PRIORITY_CHAN_5	This field defines the priority of the new frame event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.27 DC Routine Link Register 1 Channel 5 (DC_RL1_CH_5)

Address 0xBASE+0xE058068 (DC_RL1_CH_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NFIELD_START_CHAN_5								0	0	0	0	COD_NFIELD_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOF_START_CHAN_5								0	0	0	0	COD_EOF_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-302. DC Routine Link Register 1 Channel 5 (DC_RL1_CH_5)

Table 42-305. DC_RL1_CH_5 Field Descriptions

Field	Description
31–24 COD_NFIELD_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new field event resides (associated with channel #5)
23–20	Reserved.
19–16 COD_NFIELD_PRIORITY_CHAN_5	This field defines the priority of the new field event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOF_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the end of frame event resides (associated with channel #5)
7–4	Reserved.
3–0 COD_EOF_PRIORITY_CHAN_5	This field defines the priority of the end of frame event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.28 DC Routine Link Register 2 Channel 5 (DC_RL2_CH_5)

Address 0xBASE+0xE05806C (DC_RL2_CH_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_EOFIELD_START_CHAN_5								0	0	0	0	COD_EOFIELD_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOL_START_CHAN_5								0	0	0	0	COD_EOL_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-303. DC Routine Link Register 2 Channel 5 (DC_RL2_CH_5)

Table 42-306. DC_RL2_CH_5 Field Descriptions

Field	Description
31–24 COD_EOFIELD_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the end of field event resides (associated with channel #5)
23–20	Reserved.
19–16 COD_EOFIELD_PRIORITY_CHAN_5	This field defines the priority of the end of field event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOL_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the end of line event resides (associated with channel #5)

Table 42-306. DC_RL2_CH_5 Field Descriptions (continued)

Field	Description
7–4	Reserved.
3–0 COD_EOL_PRIORITY_CHAN_5	This field defines the priority of the end of line event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.29 DC Routine Link Register 3 Channel 5 (DC_RL3_CH_5)

Address 0xBASE+0xE058070 (DC_RL3_CH_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_5								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_5								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-304. DC Routine Link Register 3 Channel 5 (DC_RL3_CH_5)

Table 42-307. DC_RL3_CH_5 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #5)
23–20	Reserved.

Table 42-307. DC_RL3_CH_5 Field Descriptions (continued)

Field	Description
19–16 COD_NEW_CHAN_PRIORITY_CHAN_5	This field defines the priority of the new channel event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NEW_ADDR_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #5)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_5	This field defines the priority of the new address event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.30 DC Routine Link Register 4 Channel 5 (DC_RL4_CH_5)

Address 0xBASE+0xE058074 (DC_RL4_CH_5) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_5								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-305. DC Routine Link Register 4 Channel 5 (DC_RL4_CH_5)

Table 42-308. DC_RL4_CH_5 Field Descriptions

Field	Description
31–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_5	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #5)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_5	This field defines the priority of the new data event (associated with channel #5) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.31 DC Write Channel 6 Configuration Register (DC_WR_CH_CONF_6)

Address 0xBASE+0xE058078 (DC_WR_CH_CONF_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	PROG_START_TIME_6										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	CHAN_MASK_DEFAULT_6	PROG_CHAN_TYP_6			PROG_DISP_ID_6		PROG_DI_ID_6	W_SIZE_6	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-306. DC Write Channel 6 Configuration Register (DC_WR_CH_CONF_6)

Table 42-309. DC_WR_CH_CONF_6 Field Descriptions

Field	Description
31–27	Reserved.
26–16 PROG_ START_ TIME_6	This field defines the delay between display ‘s vertical synchronization pulse and the start time point of DC’s channel 6 window. The delay is defined in pairs of rows. It is used for tearing elimination
15–9	Reserved.
8 CHAN_ MASK_D EFAULT _6	Event mask bit for channel #6 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
7–5 PROG_ CHAN_T YP_6	This field define the mode of operation of channel #6 000 - Disable 001 - Reserved 010 - Reserved 100 - Normal mode without anti-tearing. For sync display this is the only mode allowed 101 - Normal mode with anti-tearing 110 - Reserved 111 - Additional command channel is added to the flow handled by DC channel #6
4–3 PROG_ DISP_ID _6	The field defines which one of the 4 displays is associated with channel #6. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 PROG_ DI_ID_6	This bit select the DI which a transaction associated with channel #6 can be performed to 1 - DI #1 0 - DI #0
1-0 W_SIZE _6	Word Size associated with channel #6 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.32 DC Write Channel 6 Configuration Register (DC_WR_CH_ADDR_6)

Address 0xBASE+0xE05807C (DC_WR_CH_ADDR_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_6												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_6															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-307. DC Write Channel 6 Configuration Register (DC_WR_CH_ADDR_6)

Table 42-310. DC_WR_CH_ADDR_6 Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_6	This field defines the start address within the display's memory space where the write transactions will be done to for channel #6.

42.2.3.12.33 DC Routine Link Register 0 Channel 6 (DC_RL0_CH_6)

Address 0xBASE+0xE058080 (DC_RL0_CH_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NL_START_CHAN_6								0	0	0	0	COD_NL_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NF_START_CHAN_6								0	0	0	0	COD_NF_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-308. DC Routine Link Register 0 Channel 6 (DC_RL0_CH_6)

Table 42-311. DC_RL0_CH_6 Field Descriptions

Field	Description
31–16	Reserved.
31–24 COD_NL_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new line event resides (associated with channel #6)
23–20	Reserved.
19–16 COD_NL_PRIORITY_CHAN_6	This field defines the priority of the new line event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NF_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new frame event resides (associated with channel #6)
7–4	Reserved.
3–0 COD_NF_PRIORITY_CHAN_6	This field defines the priority of the new frame event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.34 DC Routine Link Register 1 Channel 6 (DC_RL1_CH_6)

Address 0xBASE+0xE058084 (DC_RL1_CH_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NFIELD_START_CHAN_6								0	0	0	0	COD_NFIELD_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOF_START_CHAN_6								0	0	0	0	COD_EOF_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-309. DC Routine Link Register 1 Channel 6 (DC_RL1_CH_6)

Table 42-312. DC_RL1_CH_6 Field Descriptions

Field	Description
31–16	Reserved.
31–24 COD_NFIELD_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new field event resides (associated with channel #6)
23–20	Reserved.
19–16 COD_NFIELD_PRIORITY_CHAN_6	This field defines the priority of the new field event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOF_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the end of frame event resides (associated with channel #6)

Table 42-312. DC_RL1_CH_6 Field Descriptions (continued)

Field	Description
7–4	Reserved.
3–0 COD_EOF_PRIORITY_CHAN_6	This field defines the priority of the end of frame event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.35 DC Routine Link Register 2 Channel 6 (DC_RL2_CH_6)

Address 0xBASE+0xE058088 (DC_RL2_CH_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_EOFIELD_START_CHAN_6								0	0	0	0	COD_EOFIELD_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EOL_START_CHAN_6								0	0	0	0	COD_EOL_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-310. DC Routine Link Register 2 Channel 6 (DC_RL2_CH_6)

Table 42-313. DC_RL2_CH_6 Field Descriptions

Field	Description
31–24 COD_EOFIELD_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the end of field event resides (associated with channel #6)
23–20	Reserved.

Table 42-313. DC_RL2_CH_6 Field Descriptions (continued)

Field	Description
19–16 COD_EOFIELD_PRIORITY_CHAN_6	This field defines the priority of the end of field event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_EOL_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the end of line event resides (associated with channel #6)
7–4	Reserved.
3–0 COD_EOL_PRIORITY_CHAN_6	This field defines the priority of the end of line event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.36 DC Routine Link Register 3 Channel 6 (DC_RL3_CH_6)

Address 0xBASE+0xE05808C (DC_RL3_CH_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_6								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_6								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-311. DC Routine Link Register 3 Channel 6 (DC_RL3_CH_6)

Table 42-314. DC_RL3_CH_6 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #6)
23–20	Reserved.
19–16 COD_NEW_CHAN_PRIORITY_CHAN_6	This field defines the priority of the new channel event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
15–8 COD_NEW_ADDR_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #6)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_6	This field defines the priority of the new address event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.37 DC Routine Link Register 4 Channel 6 (DC_RL4_CH_6)

Address 0xBASE+0xE058090 (DC_RL4_CH_6) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_6								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_6			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-312. DC Routine Link Register 4 Channel 6 (DC_RL4_CH_6)

Table 42-315. DC_RL4_CH_6 Field Descriptions

Field	Description
31–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_6	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #6)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_6	This field defines the priority of the new data event (associated with channel #6) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.38 DC Write Channel 8 Configuration 1 Register (DC_WR_CH_CONF1_8)

Address 0xBASE+0xE058094 (DC_WR_CH_CONF1_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	MCU_DISP_ID_8		CHAN_MASK_DEFAULT_8	W_SIZE_8	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-313. DC Write Channel 8 Configuration 1 Register (DC_WR_CH_CONF1_8)

Table 42-316. DC_WR_CH_CONF1_8 Field Descriptions

Field	Description
31–5	Reserved.
4–3 MCU_DISP_ID_8	The field defines which one of the 4 displays is associated with channel #8. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 CHAN_MASK_DEFAULT_8	Event mask bit for channel #8 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
1–0 W_SIZE_8	Word Size associated with channel #8 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.39 DC Write Channel 8 Configuration 2 Register (DC_WR_CH_CONF2_8)

Address 0xBASE+0xE058098 (DC_WR_CH_CONF2_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	NEW_ADDR_SPACE_SA_8												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEW_ADDR_SPACE_SA_8															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-314. DC Write Channel 8 Configuration 2 Register (DC_WR_CH_CONF2_8)

Table 42-317. DC_WR_CH_CONF2_8 Field Descriptions

Field	Description
31–29	Reserved.
28–0 NEW_ADDR_SPACE_SA_8	Channel #8 is used for MCU direct access to the display. This field defines the base address of the second region accessible on the display

42.2.3.12.40 DC Routine Link Register 1 Channel 8 (DC_RL1_CH_8)

Address 0xBASE+0xE05809C (DC_RL1_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_ADDR_START_CHAN_W_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_W_8_0								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-315. DC Routine Link Register 1 Channel 8 (DC_RL1_CH_8)

Table 42-318. DC_RL1_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_ADDR_START_CHAN_W_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #8, second region)
23–16	Reserved.
15–8 COD_NEW_ADDR_START_CHAN_W_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #8, first region)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_8	This field defines the priority of the new address event (associated with channel #8, both regions) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.41 DC Routine Link Register 2 Channel 8 (DC_RL2_CH_8)

Address 0xBASE+0xE0580A0 (DC_RL2_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_W_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_CHAN_START_CHAN_W_8_0								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-316. DC Routine Link Register 2 Channel 8 (DC_RL2_CH_8)

Table 42-319. DC_RL2_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_W_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #8, second region)
23–16	Reserved.
15–8 COD_NEW_CHAN_START_CHAN_W_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #8, first region)
7–4	Reserved.
3–0 COD_NEW_CHAN_PRIORITY_CHAN_8	This field defines the priority of the new channel event (associated with channel #8, both regions) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.42 DC Routine Link Register 3 Channel 8 (DC_RL3_CH_8)

Address 0xBASE+0xE0580A4 (DC_RL3_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_DATA_START_CHAN_W_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_W_8_0								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_8			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-317. DC Routine Link Register 3 Channel 8 (DC_RL3_CH_8)

Table 42-320. DC_RL3_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_DATA_START_CHAN_W_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #8,second region)
23–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_W_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #8, first region)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_8	This field defines the priority of the new data event (associated with channel #8, both regions) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.43 DC Routine Link Register 4 Channel 86 (DC_RL4_CH_8)

Address 0xBASE+0xE0580A8 (DC_RL4_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_ADDR_START_CHAN_R_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_R_8_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-318. DC Routine Link Register 4 Channel 86 (DC_RL4_CH_8)

Table 42-321. DC_RL4_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_ADDR_START_CHAN_R_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #8, second region)
23–16	Reserved.
15–8 COD_NEW_ADDR_START_CHAN_R_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #8, first region)
7–0	Reserved.

42.2.3.12.44 DC Routine Link Register 5 Channel 6 (DC_RL5_CH_8)

Address 0xBASE+0xE0580AC (DC_RL5_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_R_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_CHAN_START_CHAN_R_8_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-319. DC Routine Link Register 5 Channel 6 (DC_RL5_CH_8)

Table 42-322. DC_RL5_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_R_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #8, second region)
23–16	Reserved.

Table 42-322. DC_RL5_CH_8 Field Descriptions (continued)

Field	Description
15–8 COD_NEW_CHAN_START_CHAN_R_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #8, first region)
7–0	Reserved.

42.2.3.12.45 DC Routine Link Register 6 Channel 8 (DC_RL6_CH_8)

Address 0xBASE+0xE0580B0 (DC_RL6_CH_8) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_DATA_START_CHAN_R_8_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_R_8_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-320. DC Routine Link Register 6 Channel 8 (DC_RL6_CH_8)

Table 42-323. DC_RL6_CH_8 Field Descriptions

Field	Description
31–24 COD_NEW_DATA_START_CHAN_R_8_1	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #8, second region)
23–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_R_8_0	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #8, first region)
7–0	Reserved.

42.2.3.12.46 DC Write Channel 9 Configuration 1 Register (DC_WR_CH_CONF1_9)

Address 0xBASE+0xE0580B4 (DC_WR_CH_CONF1_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	MCU_DISP_ID_9		CHAN_MASK_DEFAULT_9	W_SIZE_9	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-321. DC Write Channel 9 Configuration 1 Register (DC_WR_CH_CONF1_9)

Table 42-324. DC_WR_CH_CONF1_9 Field Descriptions

Field	Description
31–5	Reserved.
4–3 MCU_DISP_ID_9	The field defines which one of the 4 displays is associated with channel #9. 00 - display #0 01 - display #1 10 - display #2 11 - display #3
2 CHAN_MASK_DEFAULT_9	Event mask bit for channel #9 When more then one event is used during a flow (EOF, EOL, NL, NF, EOFIELD, etc.) masks all the event besides the event that is defined as the highest priority event 1 - All the events are used - no mask 0 - Only the highest priority event is used, the rest are masked
1–0 W_SIZE_9	Word Size associated with channel #9 The data coming from the IDMAC is 32bit wide. This field defines the size of the word used by the DC 00 - 8 bits are used - a 32 bit words includes 4X8bit valid words. 01 - 16 LSB bits - a 32 bit words includes 2 X16bit valid words. 10 - 24 MSB bits are used (RGB) - 8 LSB bits are ignored by the DC 11 - 32 bits are used

42.2.3.12.47 DC Write Channel 9 Configuration 2 Register (DC_WR_CH_CONF2_9)

Address 0xBASE+0xE0580B8 (DC_WR_CH_CONF2_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	NEW_ADDR_SPACE_SA_9												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NEW_ADDR_SPACE_SA_9															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-322. DC Write Channel 9 Configuration 1 Register (DC_WR_CH_CONF1_9)

Table 42-325. DC_WR_CH_CONF2_9 Field Descriptions

Field	Description
31–16	Reserved.
28–0 NEW_ADDR_SPACE_SA_9	Channel #8 is used for MCU direct access to the display. This field defines the base address of the second region accessible on the display

42.2.3.12.48 DC Routine Link Register 1 Channel 9 (DC_RL1_CH_9)

Address 0xBASE+0xE0580BC (DC_RL1_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_ADDR_START_CHAN_W_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_W_9_0								0	0	0	0	COD_NEW_ADDR_PRIORITY_CHAN_9			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-323. DC Routine Link Register 1 Channel 9 (DC_RL1_CH_9)

Table 42-326. DC_RL1_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_ADDR_START_CHAN_W_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #9, second region)
23–16	Reserved.
15–8 COD_NEW_ADDR_START_CHAN_W_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #9, first region)
7–4	Reserved.
3–0 COD_NEW_ADDR_PRIORITY_CHAN_9	This field defines the priority of the new address event (associated with channel #9, both regions) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.49 DC Routine Link Register 2 Channel 9 (DC_RL2_CH_9)

Address 0xBASE+0xE0580C0 (DC_RL2_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_W_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_CHAN_START_CHAN_W_9_0								0	0	0	0	COD_NEW_CHAN_PRIORITY_CHAN_9			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-324. DC Routine Link Register 2 Channel 9 (DC_RL2_CH_9)

Table 42-327. DC_RL2_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_W_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #9, second region)
23–16	Reserved.
15–8 COD_NEW_CHAN_START_CHAN_W_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #9, first region)
7–4	Reserved.
3–0 COD_NEW_CHAN_PRIORITY_CHAN_9	This field defines the priority of the new channel event (associated with channel #9, both region) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.50 DC Routine Link Register 3 Channel 9 (DC_RL3_CH_9)

Address 0xBASE+0xE0580C4 (DC_RL3_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_DATA_START_CHAN_W_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_W_9_0								0	0	0	0	COD_NEW_DATA_PRIORITY_CHAN_9			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-325. DC Routine Link Register 3 Channel 9 (DC_RL3_CH_9)

Table 42-328. DC_RL3_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_DATA_START_CHAN_W_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #9, second region)
23–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_W_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #9, first region)
7–4	Reserved.
3–0 COD_NEW_DATA_PRIORITY_CHAN_9	This field defines the priority of the new data event (associated with channel #9, both regions) 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)

42.2.3.12.51 DC Routine Link Register 4 Channel 6 (DC_RL4_CH_9)

Address 0xBASE+0xE0580C8 (DC_RL4_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_ADDR_START_CHAN_R_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_ADDR_START_CHAN_R_9_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-326. DC Routine Link Register 4 Channel 6 (DC_RL4_CH_9)

Table 42-329. DC_RL4_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_ADDR_START_CHAN_R_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #9, second region)
23–16	Reserved.
15–8 COD_NEW_ADDR_START_CHAN_R_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new address event resides (associated with channel #9, first region)
7–0	Reserved.

42.2.3.12.52 DC Routine Link Register 5 Channel 6 (DC_RL5_CH_9)

Address 0xBASE+0xE0580CC (DC_RL5_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_CHAN_START_CHAN_R_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_CHAN_START_CHAN_R_9_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-327. DC Routine Link Register 5 Channel 6 (DC_RL5_CH_9)

Table 42-330. DC_RL5_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_CHAN_START_CHAN_R_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #9, second region)
23–16	Reserved.

Table 42-330. DC_RL5_CH_9 Field Descriptions (continued)

Field	Description
15–8 COD_NEW_CHAN_START_CHAN_R_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new channel event resides (associated with channel #9, first region)
7–0	Reserved.

42.2.3.12.53 DC Routine Link Register 6 Channel 9 (DC_RL3_CH_9)

Address 0xBASE+0xE0580D0 (DC_RL6_CH_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COD_NEW_DATA_START_CHAN_R_9_1								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_NEW_DATA_START_CHAN_R_9_0								0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-328. DC Routine Link Register 6 Channel 9 (DC_RL3_CH_9)

Table 42-331. DC_RL6_CH_9 Field Descriptions

Field	Description
31–24 COD_NEW_DATA_START_CHAN_R_9_1	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #9, second region)
23–16	Reserved.
15–8 COD_NEW_DATA_START_CHAN_R_9_0	This field is a pointer to the address within the microcode memory where the routine that handles the new data event resides (associated with channel #9, first region)
7–0	Reserved.

42.2.3.12.54 DC General Register (DC_GEN)

Address 0xBASE+0xE0580D4 (DC_GEN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	DC_BK_EN	DC_BKDIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	DC_CH5_TYPE	SYNC_PRIORITY_1	SYNC_PRIORITY_5	MASK4CHAN_5	MASK_EN	0	SYNC_1_6		0
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Figure 42-329. DC General Register (DC_GEN)

Table 42-332. DC_GEN Field Descriptions

Field	Description
31–25	Reserved.
24 DC_BK_EN	Cursor blinking enable 1- blinking is enabled 0 - blinking is disabled
23–16 DC_BKDIV	Blinking Rate This field defines the blinking rate. The blinking occurs every N-th frame While N is defined by DC_BKDIV
15–9	Reserved.
8 DC_CH5_TYPE	Channel 5 is used for synchronous flow. When this channel is used for accessing asynchronous display that is activated in a synchronous way 1 - Enable the asynchronous interface via channel 5 0 - normal mode, synchronous flow via channel 5
7 SYNC_PRIORITY_1	When 2 sync flows are running, this bit sets the priority of channel #1. This bit should be 1 - high Priority 0 - low Priority both SYNC_PRIORITY_5 and SYNC_PRIORITY_1 should not have the value of 0
6 SYNC_PRIORITY_5	When 2 sync flows are running, this bit sets the priority of channel #5. This bit should be 1 - high priority 0 - low Priority both SYNC_PRIORITY_5 and SYNC_PRIORITY_1 should not have the value of 0
5 MASK4CHAN_5	Sync flow can be associated with a mask channel. Only one sync flow can have a mask. 1 - mask channel is associated to the sync flow via DP 0 - mask channel is associated to the sync flow via DC (without DP) This bit is ignored if MASK_EN is clear

Table 42-332. DC_GEN Field Descriptions (continued)

Field	Description
4 MASK_EN	Enable of the mask channel 1 - mask channel is enabled 0 - mask channel is disabled
3	Reserved
2–1 SYNC_1_6	This field 00 - Channel 1 of the DC handles async flow 01 - Illegal 10 - Channel 1 of the DC handles sync flow 11 - illegal
0	Reserved

42.2.3.12.55 DC Display Configuration 1 Register 0 (DC_DISP_CONF1_0)

Address 0xBASE+0xE0580D8 (DC_DISP_CONF1_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DISP_RD_VALUE_PTR_0	MCU_ACC_LB_MASK_0	ADDR_BE_L_INC_0		ADDR_INCREMENT_0		DISP_TYP_0	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

Figure 42-330. DC Display Configuration 1 Register 0 (DC_DISP_CONF1_0)

Table 42-333. DC_DISP_CONF1_0 Field Descriptions

Field	Description
31–8	Reserved.
7 DISP_RD_VALUE_PTR_0	When the display works in wait for status mode. The IPUv3EX polls the display and compare the value to the value stored on the status DI_READ_DATA_ACK_VALUE and mask it with DI_READ_DATA_MASK. The DC holds 2 sets of mask and value. This field holds a pointer to the corresponding set of mask and value. 1 - DI_READ_DATA_ACK_VALUE_1 & DI_READ_DATA_MASK_1 are used for display 0 0 - DI_READ_DATA_ACK_VALUE_0 & DI_READ_DATA_MASK_0 are used for display 0

Table 42-333. DC_DISP_CONF1_0 Field Descriptions (continued)

Field	Description
6 MCU_ACC_LB_MASK_0	The DC compares between the current access to the a calculated address. The calculated address is the next consecutive address following the last address. This bit defines the comparing mode 1 - The 2 addresses are fully compared 0 - The 2 addresses are compared, but the ADDR[0] bit of the new address is ignored
5–4 ADDR_BE_L_INC_0	This bits define the increment mode when the latest access was done with some of the byte enable signals are low, in that case different increment should be done instead of the normal auto increment of the address 00 - No increment 01 - Increment by 1 10 - Increment by 2 11 - Increment by 3 IF MCU_ACC_LB_MASK_0 is 0 then only 00 and 10 values are allowed.
3–2 ADDR_INCREMENT_0	This field is the increment step for auto increment mode 00 - Increment the address by 1 01 - Increment the address by 2 10 - Increment the address by 3 11 - Increment the address by 4
1–0 DISP_TYP_0	This field defines the type of the display 00 - Serial accesses display 01 - Reserved 10 - parallel display, without byte_enable support 11 - parallel display, with byte_enable support

42.2.3.12.56 DC Display Configuration 1 Register 1 (DC_DISP_CONF1_1)

Address 0xBASE+0xE0580DC (DC_DISP_CONF1_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DISP_RD_VALUE_PTR_1	MCU_ACC_LB_MASK_1	ADDR_BE_L_INC_1		ADDR_INCREMENT_1		DISP_TYP_1	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

Figure 42-331. DC Display Configuration 1 Register 1 (DC_DISP_CONF1_1)

Table 42-334. DC_DISP_CONF1_1 Field Descriptions

Field	Description
31–8	Reserved.
7 DISP_RD_VALUE_PTR_1	When the display works in wait for status mode. The IPUv3EX polls the display and compare the value to the value stored on the status DI_READ_DATA_ACK_VALUE and mask it with DI_READ_DATA_MASK. The DC holds 2 sets of mask and value. This field holds a pointer to the corresponding set of mask and value. 1 - DI_READ_DATA_ACK_VALUE_1 & DI_READ_DATA_MASK_1 are used for display 1 0- DI_READ_DATA_ACK_VALUE_0 & DI_READ_DATA_MASK_0 are used for display 1
6 MCU_ACC_LB_MASK_1	The DC compares between the current access to the a calculated address. The calculated address is the next consecutive address following the last address. This bit defines the comparing mode 1 - The 2 addresses are fully compared 0 - The 2 addresses are compared, but the ADDR[0] bit of the new address is ignored
5–4 ADDR_BE_L_INC_1	This bits define the increment mode when the latest access was done with some of the byte enable signals are low, in that case different increment should be done instead of the normal auto increment of the address 00 - No increment 01 - Increment by 1 10 - Increment by 2 11 - Increment by 3 IF MCU_ACC_LB_MASK_1 is 0 then only 00 and 10 values are allowed.
3–2 ADDR_INCREMENT_1	This field is the increment step for auto increment mode 00 - Increment the address by 1 01 - Increment the address by 2 10 - Increment the address by 3 11 - Increment the address by 4
1–0 DISP_TYP_1	This field defines the type of the display 00 - Serial accesses display 01 - Reserved 10 - parallel display, without byte_enable support 11 - parallel display, with byte_enable support

42.2.3.12.57 DC Display Configuration 1 Register 2 (DC_DISP_CONF1_2)

Address 0xBASE+0xE0580E0 (DC_DISP_CONF1_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DISP_RD_VALUE_PTR_2	MCU_ACC_LB_MASK_2	ADDR_BE_L_INC_2		ADDR_INCREMENT_2		DISP_TYP_2	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

Figure 42-332. DC Display Configuration 1 Register 2 (DC_DISP_CONF1_2)

Table 42-335. DC_DISP_CONF1_2 Field Descriptions

Field	Description
31–8	Reserved.
7 DISP_RD_VALUE_PTR_2	When the display works in wait for status mode. The IPUv3EX polls the display and compare the value to the value stored on the status DI_READ_DATA_ACK_VALUE and mask it with DI_READ_DATA_MASK. The DC holds 2 sets of mask and value. This field holds a pointer to the corresponding set of mask and value. 1 - DI_READ_DATA_ACK_VALUE_1 & DI_READ_DATA_MASK_1 are used for display 2 0- DI_READ_DATA_ACK_VALUE_0 & DI_READ_DATA_MASK_0 are used for display 2
6 MCU_ACC_LB_MASK_2	The DC compares between the current access to the a calculated address. The calculated address is the next consecutive address following the last address. This bit defines the comparing mode 1 - The 2 addresses are fully compared 0 - The 2 addresses are compared, but the ADDR[0] bit of the new address is ignored
5–4 ADDR_BE_L_INC_2	This bits define the increment mode when the latest access was done with some of the byte enable signals are low, in that case different increment should be done instead of the normal auto increment of the address 00 - No increment 01 - Increment by 1 10 - Increment by 2 11 - Increment by 3 IF MCU_ACC_LB_MASK_2 is 0 then only 00 and 10 values are allowed.

Table 42-335. DC_DISP_CONF1_2 Field Descriptions (continued)

Field	Description
3–2 ADDR_INCREMENT_2	This field is the increment step for auto increment mode 00 - Increment the address by 1 01 - Increment the address by 2 10 - Increment the address by 3 11 - Increment the address by 4
1–0 DISP_TYP_2	This field defines the type of the display 00 - Serial accesses display 01 - Reserved 10 - parallel display, without byte_enable support 11 - parallel display, with byte_enable support

42.2.3.12.58 DC Display Configuration 1 Register 3 (DC_DISP_CONF1_3)

Address 0xBASE+0xE0580E4 (DC_DISP_CONF1_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DISP_RD_VALUE_PTR_3	MCU_ACC_LB_MASK_3	ADDR_BE_L_INC_3		ADDR_INCREMENT_3		DISP_TYP_3	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

Figure 42-333. DC Display Configuration 1 Register 3 (DC_DISP_CONF1_3)

Table 42-336. DC_DISP_CONF1_3 Field Descriptions

Field	Description
31–8	Reserved.
7 DISP_RD_VALUE_PTR_3	When the display works in wait for status mode. The IPUv3EX polls the display and compare the value to the value stored on the status DI_READ_DATA_ACK_VALUE and mask it with DI_READ_DATA_MASK. The DC holds 2 sets of mask and value. This field holds a pointer to the corresponding set of mask and value. 1 - DI_READ_DATA_ACK_VALUE_1 & DI_READ_DATA_MASK_1 are used for display 3 0- DI_READ_DATA_ACK_VALUE_0 & DI_READ_DATA_MASK_0 are used for display 3

Table 42-336. DC_DISP_CONF1_3 Field Descriptions (continued)

Field	Description
6 MCU_ACC_LB_MASK_3	The DC compares between the current access to the a calculated address. The calculated address is the next consecutive address following the last address. This bit defines the comparing mode 1 - The 2 addresses are fully compared 0 - The 2 addresses are compared, but the ADDR[0] bit of the new address is ignored
5–4 ADDR_BE_L_INC_3	This bits define the increment mode when the latest access was done with some of the byte enable signals are low, in that case different increment should be done instead of the normal auto increment of the address 00 - No increment 01 - Increment by 1 10 - Increment by 2 11 - Increment by 3 IF MCU_ACC_LB_MASK_3 is 0 then only 00 and 10 values are allowed.
3–2 ADDR_INCREMENT_3	This field is the increment step for auto increment mode 00 - Increment the address by 1 01 - Increment the address by 2 10 - Increment the address by 3 11 - Increment the address by 4
1–0 DISP_TYP_3	This field defines the type of the display 00 - Serial accesses display 01 - Reserved 10 - parallel display, without byte_enable support 11 - parallel display, with byte_enable support

42.2.3.12.59 DC Display Configuration 2 Register 0 (DC_DISP_CONF_0)

Address 0xBASE+0xE0580E8 (DC_DISP_CONF2_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	SL_0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SL_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-334. DC Display Configuration 2 Register 0 (DC_DISP_CONF_0)

Table 42-337. DC_DISP_CONF_0 Field Descriptions

Field	Description
31–29	Reserved.
28–0 SL_0	Stride line of display 0

42.2.3.12.60 DC Display Configuration 2 Register 1 (DC_DISP_CONF2_1)

Address 0xBASE+0xE0580EC (DC_DISP_CONF2_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	SL_1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SL_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-335. DC Display Configuration 2 Register 1 (DC_DISP_CONF2_1)

Table 42-338. DC_DISP_CONF2_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 SL_1	Stride line of display 1

42.2.3.12.61 DC Display Configuration 2 Register 2 (DC_DISP_CONF2_2)

Address 0xBASE+0xE0580F0 (DC_DISP_CONF2_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	SL_2												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SL_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-336. DC Display Configuration 2 Register 2 (DC_DISP_CONF2_2)

Table 42-339. DC_DISP_CONF2_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 SL_2	Stride line of display 2

42.2.3.12.62 DC Display Configuration 2 Register 3 (DC_DISP_CONF2_3)

Address 0xBASE+0xE0580F4 (DC_DISP_CONF2_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	SL_3												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SL_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-337. DC Display Configuration 2 Register 3 (DC_DISP_CONF2_3)

Table 42-340. DC_DISP_CONF2_3 Field Descriptions

Field	Description
31–29	Reserved.
28–0 SL_3	Stride line of display 3

42.2.3.12.63 DC DI0 Configuration Register 1 (DC_DI0_CONF_1)

Address 0xBASE+0xE0580F8 (DC_DI0_CONF_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DI_READ_DATA_MASK_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI_READ_DATA_MASK_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-338. DC DI0 Configuration Register 1 (DC_DI0_CONF_1)

Table 42-341. DC_DI0_CONF_1 Field Descriptions

Field	Description
31–0 DI_READ_DATA_MASK_0	This field defines the mask value of the data read from the display.

42.2.3.12.64 DC DI0 Configuration Register 2 (DC_DI0_CONF_2)

Address 0xBASE+0xE0580FC (DC_DI0_CONF_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI_READ_DATA_ACK_VALUE_0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI_READ_DATA_ACK_VALUE_0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-339. DC DI0 Configuration Register 2 (DC_DI0_CONF_2)

Table 42-342. DC_DI0_CONF_2 Field Descriptions

Field	Description
31–0 DI_READ_DATA_ACK_VALUE_0	This is the expected data to be read from the display. The value reads from the display is anded with the DI_READ_DATA_MASK_0 and compared with the DI_READ_DATA_ACK_VALUE_0. This field is used for the READ_STATUS task of the DC

42.2.3.12.65 DC DI1 Configuration Register 1 (DC_DI1_CONF_1)

Address 0xBASE+0xE058100 (DC_DI1_CONF_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI_READ_DATA_MASK_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI_READ_DATA_MASK_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-340. DC DI1 Configuration Register 1 (DC_DI1_CONF_1)

Table 42-343. DC_DI1_CONF_1 Field Descriptions

Field	Description
31–0 DI_READ_DATA_MASK_1	This field defines the mask value of the data read from the display.

42.2.3.12.66 DC DI1 Configuration Register 2 (DC_DI1_CONF_2)

Address 0xBASE+0xE058104 (DC_DI1_CONF_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DI_READ_DATA_ACK_VALUE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DI_READ_DATA_ACK_VALUE_1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-341. DC DI1 Configuration Register 2 (DC_DI1_CONF_2)

Table 42-344. DC_DI1_CONF_2 Field Descriptions

Field	Description
31–0 DI_READ_DATA_ACK_VALUE_1	This is the expected data to be read from the display. The value reads from the display is anded with the DI_READ_DATA_MASK_1 and compared with the DI_READ_DATA_ACK_VALUE_1 This field is used for the READ_STATUS task of the DC

42.2.3.12.67 DC Mapping Configuration Register 0 (DC_MAP_CONF_0)

Address 0xBASE+0xE058108 (DC_MAP_CONF_0) Access: User read/write

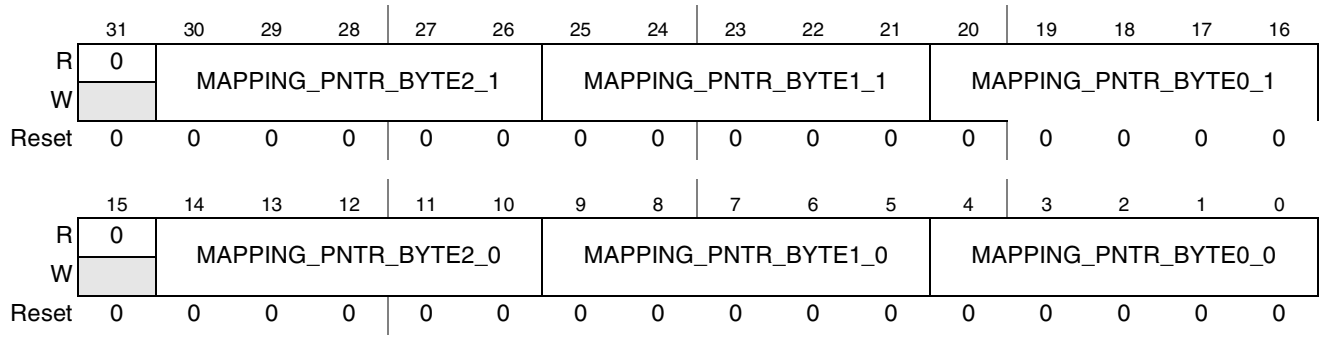


Figure 42-342. DC Mapping Configuration Register 0 (DC_MAP_CONF_0)

Table 42-345. DC_MAP_CONF_0 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_1	Mapping pointer #1 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_1	Mapping pointer #1 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_1	Mapping pointer #1 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_0	Mapping pointer #0 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_0	Mapping pointer #0 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_0	Mapping pointer #0 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.68 DC Mapping Configuration Register 1 (DC_MAP_CONF_1)

Address 0xBASE+0xE05810C (DC_MAP_CONF_1) Access: User read/write

	31	30–26	25–21	20–16	15	14–10	9–5	4–0
R	0	MAPPING_PNTR_BYTE2_3	MAPPING_PNTR_BYTE1_3	MAPPING_PNTR_BYTE0_3	0	MAPPING_PNTR_BYTE2_2	MAPPING_PNTR_BYTE1_2	MAPPING_PNTR_BYTE0_2
W								
Reset	0	0	0	0	0	0	0	0

Figure 42-343. DC Mapping Configuration Register 1 (DC_MAP_CONF_1)

Table 42-346. DC_MAP_CONF_1 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_3	Mapping pointer #3 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_3	Mapping pointer #3 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_3	Mapping pointer #3 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_1	Mapping pointer #1 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2

Table 42-346. DC_MAP_CONF_1 Field Descriptions (continued)

Field	Description
9–5 MAPPING_PNTR_BYTE1_1	Mapping pointer #1 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_1	Mapping pointer #1 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.69 DC Mapping Configuration Register 2 (DC_MAP_CONF_2)

Address 0xBASE+0xE058110 (DC_MAP_CONF_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_5							MAPPING_PNTR_BYTE1_5				MAPPING_PNTR_BYTE0_5			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_4							MAPPING_PNTR_BYTE1_4				MAPPING_PNTR_BYTE0_4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-344. DC Mapping Configuration Register 2 (DC_MAP_CONF_2)

Table 42-347. DC_MAP_CONF_2 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_5	Mapping pointer #5 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_5	Mapping pointer #5 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_5	Mapping pointer #5 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.

Table 42-347. DC_MAP_CONF_2 Field Descriptions (continued)

Field	Description
14–10 MAPPING_PNTR_BYTE2_4	Mapping pointer #4 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_4	Mapping pointer #4 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_4	Mapping pointer #4 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.70 DC Mapping Configuration Register 3 (DC_MAP_CONF_3)

Address 0xBASE+0xE058114 (DC_MAP_CONF_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_7					MAPPING_PNTR_BYTE1_7					MAPPING_PNTR_BYTE0_7				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_6					MAPPING_PNTR_BYTE1_6					MAPPING_PNTR_BYTE0_6				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-345. DC Mapping Configuration Register 3 (DC_MAP_CONF_3)

Table 42-348. DC_MAP_CONF_3 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_7	Mapping pointer #7 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_7	Mapping pointer #7 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1

Table 42-348. DC_MAP_CONF_3 Field Descriptions (continued)

Field	Description
20–16 MAPPING_PNTR_BYTE0_7	Mapping pointer #7 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_6	Mapping pointer #6 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_6	Mapping pointer #6 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_6	Mapping pointer #6 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.71 DC Mapping Configuration Register 4 (DC_MAP_CONF_4)

Address 0xBASE+0xE058118 (DC_MAP_CONF_4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_9					MAPPING_PNTR_BYTE1_9					MAPPING_PNTR_BYTE0_9				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_8					MAPPING_PNTR_BYTE1_8					MAPPING_PNTR_BYTE0_8				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-346. DC Mapping Configuration Register 4 (DC_MAP_CONF_4)

Table 42-349. DC_MAP_CONF_4 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_9	Mapping pointer #9 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_1	Mapping pointer #9 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_9	Mapping pointer #9 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_8	Mapping pointer #8 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_8	Mapping pointer #8 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_8	Mapping pointer #8 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.72 DC Mapping Configuration Register 5 (DC_MAP_CONF_5)

Address 0xBASE+0xE05811C (DC_MAP_CONF_5) Access: User read/write

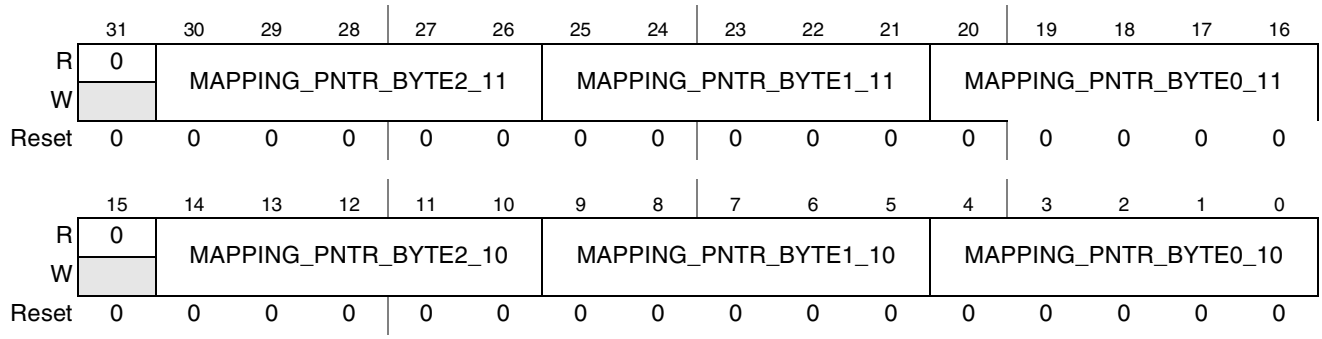


Figure 42-347. DC Mapping Configuration Register 5 (DC_MAP_CONF_5)

Table 42-350. DC_MAP_CONF_5 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_11	Mapping pointer #11 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_11	Mapping pointer #11 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_11	Mapping pointer #11 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_10	Mapping pointer #10 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_10	Mapping pointer #10 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_10	Mapping pointer #10 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.73 DC Mapping Configuration Register 6 (DC_MAP_CONF_6)

Address 0xBASE+0xE058120 (DC_MAP_CONF_6) Access: User read/write

	31	30–26	25–21	20–16	15	14–10	9–5	4–0
R	0	MAPPING_PNTR_BYTE2_13	MAPPING_PNTR_BYTE1_13	MAPPING_PNTR_BYTE0_13	0	MAPPING_PNTR_BYTE2_12	MAPPING_PNTR_BYTE1_12	MAPPING_PNTR_BYTE0_12
W								
Reset	0	0	0	0	0	0	0	0

Figure 42-348. DC Mapping Configuration Register 6 (DC_MAP_CONF_6)

Table 42-351. DC_MAP_CONF_6 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_13	Mapping pointer #13 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_13	Mapping pointer #13 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_13	Mapping pointer #13 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_12	Mapping pointer #12 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2

Table 42-351. DC_MAP_CONF_6 Field Descriptions (continued)

Field	Description
9–5 MAPPING_PNTR_BYTE1_12	Mapping pointer #12 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_12	Mapping pointer #12 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.74 DC Mapping Configuration Register 7 (DC_MAP_CONF_7)

Address 0xBASE+0xE058124 (DC_MAP_CONF_7) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_15					MAPPING_PNTR_BYTE1_15					MAPPING_PNTR_BYTE0_15				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_14					MAPPING_PNTR_BYTE1_14					MAPPING_PNTR_BYTE0_14				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-349. DC Mapping Configuration Register 7 (DC_MAP_CONF_7)

Table 42-352. DC_MAP_CONF_7 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_15	Mapping pointer #15 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_15	Mapping pointer #15 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_15	Mapping pointer #15 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

Table 42-352. DC_MAP_CONF_7 Field Descriptions (continued)

Field	Description
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_14	Mapping pointer #14 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_14	Mapping pointer #14 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_14	Mapping pointer #14 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.75 DC Mapping Configuration Register 8 (DC_MAP_CONF_8)

Address 0xBASE+0xE058128 (DC_MAP_CONF_8) Access: User read/write

	31	30–26	25	24–20	19–16
R	0	MAPPING_PNTR_BYTE2_17	MAPPING_PNTR_BYTE1_17	MAPPING_PNTR_BYTE0_17	
W					
Reset	0	0	0	0	0

	15	14–10	9	8–4	3–0
R	0	MAPPING_PNTR_BYTE2_16	MAPPING_PNTR_BYTE1_16	MAPPING_PNTR_BYTE0_16	
W					
Reset	0	0	0	0	0

Figure 42-350. DC Mapping Configuration Register 8 (DC_MAP_CONF_8)

Table 42-353. DC_MAP_CONF_8 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_17	Mapping pointer #17 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2

Table 42-353. DC_MAP_CONF_8 Field Descriptions (continued)

Field	Description
25–21 MAPPING_PNTR_BYTE1_17	Mapping pointer #17 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_17	Mapping pointer #17 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_16	Mapping pointer #16 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_16	Mapping pointer #16 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_16	Mapping pointer #16 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.76 DC Mapping Configuration Register 9 (DC_MAP_CONF_9)

Address 0xBASE+0xE05812C (DC_MAP_CONF_9) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_19					MAPPING_PNTR_BYTE1_19					MAPPING_PNTR_BYTE0_19				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_18					MAPPING_PNTR_BYTE1_18					MAPPING_PNTR_BYTE0_18				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-351. DC Mapping Configuration Register 9 (DC_MAP_CONF_9)

Table 42-354. DC_MAP_CONF_9 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_19	Mapping pointer #19 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_19	Mapping pointer #19 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_19	Mapping pointer #19 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_18	Mapping pointer #18 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_18	Mapping pointer #18 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_18	Mapping pointer #18 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.77 DC Mapping Configuration Register 10 (DC_MAP_CONF_10)

Address 0xBASE+0xE058130 (DC_MAP_CONF_10) Access: User read/write

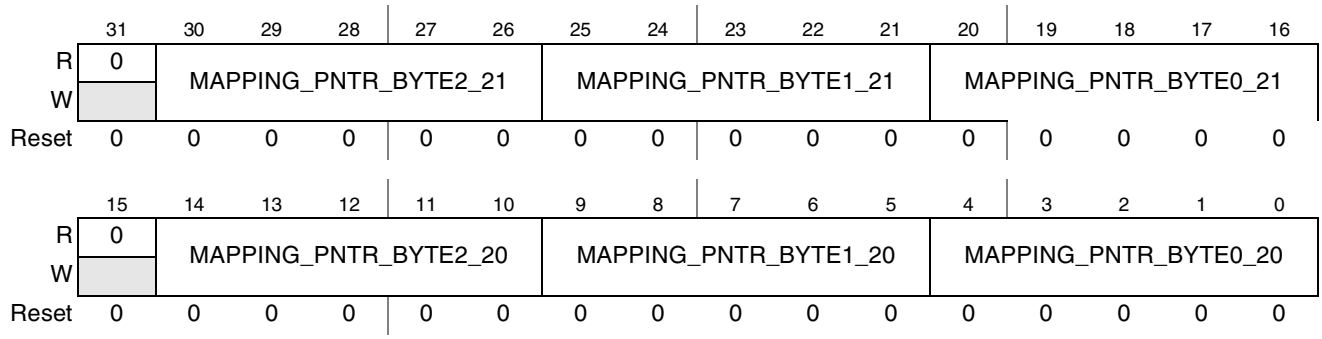


Figure 42-352. DC Mapping Configuration Register 10 (DC_MAP_CONF_10)

Table 42-355. DC_MAP_CONF_10 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_21	Mapping pointer #21 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_21	Mapping pointer #21 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_21	Mapping pointer #21 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_20	Mapping pointer #20 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_20	Mapping pointer #20 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_20	Mapping pointer #20 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.78 DC Mapping Configuration Register 11 (DC_MAP_CONF_11)

Address 0xBASE+0xE058134 (DC_MAP_CONF_11) Access: User read/write

	31	30–26	25–21	20–16	15	14–10	9–5	4–0
R	0	MAPPING_PNTR_BYTE2_23	MAPPING_PNTR_BYTE1_23	MAPPING_PNTR_BYTE0_23	0	MAPPING_PNTR_BYTE2_22	MAPPING_PNTR_BYTE1_22	MAPPING_PNTR_BYTE0_22
W								
Reset	0	0	0	0	0	0	0	0

Figure 42-353. DC Mapping Configuration Register 11 (DC_MAP_CONF_11)

Table 42-356. DC_MAP_CONF_11 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_23	Mapping pointer #23 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_23	Mapping pointer #23 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_23	Mapping pointer #23 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_22	Mapping pointer #22 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2

Table 42-356. DC_MAP_CONF_11 Field Descriptions (continued)

Field	Description
9–5 MAPPING_PNTR_BYTE1_22	Mapping pointer #22 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_22	Mapping pointer #22 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.79 DC Mapping Configuration Register 12 (DC_MAP_CONF_12)

Address 0xBASE+0xE058138 (DC_MAP_CONF_12) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_25					MAPPING_PNTR_BYTE1_25					MAPPING_PNTR_BYTE0_25				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_24					MAPPING_PNTR_BYTE1_24					MAPPING_PNTR_BYTE0_24				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-354. DC Mapping Configuration Register 12 (DC_MAP_CONF_12)

Table 42-357. DC_MAP_CONF_12 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_25	Mapping pointer #25 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_25	Mapping pointer #25 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_25	Mapping pointer #25 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

Table 42-357. DC_MAP_CONF_12 Field Descriptions (continued)

Field	Description
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_24	Mapping pointer #24 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_24	Mapping pointer #24 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_24	Mapping pointer #24 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.80 DC Mapping Configuration Register 13 (DC_MAP_CONF_13)

Address 0xBASE+0xE05813C (DC_MAP_CONF_13) Access: User read/write

	31	30–26	25–24	23–21	20–16	15	14–10	9–8	7–5	4–0
R	0	MAPPING_PNTR_BYTE2_27	MAPPING_PNTR_BYTE1_27		MAPPING_PNTR_BYTE0_27	0	MAPPING_PNTR_BYTE2_26	MAPPING_PNTR_BYTE1_26		MAPPING_PNTR_BYTE0_26
W										
Reset	0	0	0	0	0	0	0	0	0	0

Figure 42-355. DC Mapping Configuration Register 13 (DC_MAP_CONF_13)

Table 42-358. DC_MAP_CONF_13 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_27	Mapping pointer #27 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2

Table 42-358. DC_MAP_CONF_13 Field Descriptions (continued)

Field	Description
25–21 MAPPING_PNTR_BYTE1_27	Mapping pointer #27 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_27	Mapping pointer #27 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_26	Mapping pointer #26 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_26	Mapping pointer #26 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_26	Mapping pointer #26 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.81 DC Mapping Configuration Register 14 (DC_MAP_CONF_14)

Address 0xBASE+0xE058140 (DC_MAP_CONF_14) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	MAPPING_PNTR_BYTE2_29					MAPPING_PNTR_BYTE1_29					MAPPING_PNTR_BYTE0_29				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MAPPING_PNTR_BYTE2_28					MAPPING_PNTR_BYTE1_28					MAPPING_PNTR_BYTE0_28				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-356. DC Mapping Configuration Register 14 (DC_MAP_CONF_14)

Table 42-359. DC_MAP_CONF_14 Field Descriptions

Field	Description
31	Reserved.
30–26 MAPPING_PNTR_BYTE2_29	Mapping pointer #29 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
25–21 MAPPING_PNTR_BYTE1_29	Mapping pointer #29 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
20–16 MAPPING_PNTR_BYTE0_29	Mapping pointer #29 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0
15	Reserved.
14–10 MAPPING_PNTR_BYTE2_28f	Mapping pointer #28 for Byte 2 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #2
9–5 MAPPING_PNTR_BYTE1_28	Mapping pointer #28 for Byte 1 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #1
4–0 MAPPING_PNTR_BYTE2_28	Mapping pointer #28 for Byte 0 This field is a pointer to the set of MD_OFFSET_<i> and MD_MASK_<i> used for mapping byte #0

42.2.3.12.82 DC Mapping Configuration Register 15 (DC_MAP_CONF_15)

Address 0xBASE+0xE058144 (DC_MAP_CONF_15) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_1					MD_MASK_1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_0					MD_MASK_0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-357. DC Mapping Configuration Register 23 (DC_MAP_CONF_23)

Table 42-360. DC_MAP_CONF_15 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_1	Mapping unit's offset parameter #1 This field defines the offset parameter #1 within the 24bit word coming from the DC.
23–16 MD_MASK_1	Mapping unit's mask value #1 This field defines the mask value #1 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_0	Mapping unit's offset parameter #0 This field defines the offset parameter #0 within the 24bit word coming from the DC.
7–0 MD_MASK_0	Mapping unit's mask value #0 This field defines the mask value #0 within the 8bit word coming from the DC.

42.2.3.12.83 DC Mapping Configuration Register 16 (DC_MAP_CONF_16)

Address 0xBASE+0xE058148 (DC_MAP_CONF_16) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_3					MD_MASK_3							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_2					MD_MASK_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-358. DC Mapping Configuration Register 16 (DC_MAP_CONF_16)

Table 42-361. DC_MAP_CONF_16 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_3	Mapping unit's offset parameter #3 This field defines the offset parameter #3 within the 24bit word coming from the DC.
23–16 MD_MASK_3	Mapping unit's mask value #3 This field defines the mask value #3 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_2	Mapping unit's offset parameter #2 This field defines the offset parameter #2 within the 24bit word coming from the DC.
7–0 MD_MASK_0	Mapping unit's mask value #2 This field defines the mask value #2 within the 8bit word coming from the DC.
31–2	Reserved.

42.2.3.12.84 DC Mapping Configuration Register 17 (DC_MAP_CONF_17)

Address 0xBASE+0xE05814C (DC_MAP_CONF_17) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_5					MD_MASK_5							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_4					MD_MASK_4							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-359. DC Mapping Configuration Register 17 (DC_MAP_CONF_17)

Table 42-362. DC_MAP_CONF_17 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_5	Mapping unit's offset parameter #5 This field defines the offset parameter #5 within the 24bit word coming from the DC.
23–16 MD_MASK_5	Mapping unit's mask value #5 This field defines the mask value #5 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_4	Mapping unit's offset parameter #4 This field defines the offset parameter #4 within the 24bit word coming from the DC.
7–0 MD_MASK_4	Mapping unit's mask value #4 This field defines the mask value #4 within the 8bit word coming from the DC.

42.2.3.12.85 DC Mapping Configuration Register 18 (DC_MAP_CONF_18)

Address 0xBASE+0xE058150 (DC_MAP_CONF_18) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_7					MD_MASK_7							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_6					MD_MASK_6							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-360. DC Mapping Configuration Register 18 (DC_MAP_CONF_18)

Table 42-363. DC_MAP_CONF_18 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_7	Mapping unit's offset parameter #7 This field defines the offset parameter #7 within the 24bit word coming from the DC.
23–16 MD_MASK_7	Mapping unit's mask value #7 This field defines the mask value #7 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_6	Mapping unit's offset parameter #6 This field defines the offset parameter #6 within the 24bit word coming from the DC.
7–0 MD_MASK_6	Mapping unit's mask value #6 This field defines the mask value #6 within the 8bit word coming from the DC.

42.2.3.12.86 DC Mapping Configuration Register 19 (DC_MAP_CONF_19)

Address 0xBASE+0xE058154 (DC_MAP_CONF_19) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_9					MD_MASK_9							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_8					MD_MASK_8							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-361. DC Mapping Configuration Register 19 (DC_MAP_CONF_19)

Table 42-364. DC_MAP_CONF_19 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_9	Mapping unit's offset parameter #9 This field defines the offset parameter #9 within the 24bit word coming from the DC.
23–16 MD_MASK_9	Mapping unit's mask value #9 This field defines the mask value #9 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_8	Mapping unit's offset parameter #8 This field defines the offset parameter #8 within the 24bit word coming from the DC.
7–0 MD_MASK_8	Mapping unit's mask value #8 This field defines the mask value #8 within the 8bit word coming from the DC.

42.2.3.12.87 DC Mapping Configuration Register 20 (DC_MAP_CONF_20)

Address 0xBASE+0xE058158 (DC_MAP_CONF_20) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_11					MD_MASK_11							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_10					MD_MASK_10							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-362. DC Mapping Configuration Register 20 (DC_MAP_CONF_20)

Table 42-365. DC_MAP_CONF_20 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_11	Mapping unit's offset parameter #11 This field defines the offset parameter #11 within the 24bit word coming from the DC.
23–16 MD_MASK_11	Mapping unit's mask value #11 This field defines the mask value #11 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_10	Mapping unit's offset parameter #10 This field defines the offset parameter #10 within the 24bit word coming from the DC.
7–0 MD_MASK_10	Mapping unit's mask value #10 This field defines the mask value #10 within the 8bit word coming from the DC.

42.2.3.12.88 DC Mapping Configuration Register 21 (DC_MAP_CONF_21)

Address 0xBASE+0xE05815C (DC_MAP_CONF_21) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_13					MD_MASK_13							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_12					MD_MASK_12							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-363. DC Mapping Configuration Register 21 (DC_MAP_CONF_21)

Table 42-366. DC_MAP_CONF_21 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_13	Mapping unit's offset parameter #13 This field defines the offset parameter #13 within the 24bit word coming from the DC.
23–16 MD_MASK_13	Mapping unit's mask value #13 This field defines the mask value #13 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_12	Mapping unit's offset parameter #12 This field defines the offset parameter #12 within the 24bit word coming from the DC.
7–0 MD_MASK_12	Mapping unit's mask value #12 This field defines the mask value #12 within the 8bit word coming from the DC.

42.2.3.12.89 DC Mapping Configuration Register 22 (DC_MAP_CONF_22)

Address 0xBASE+0xE058160 (DC_MAP_CONF_22) Access: User read/write

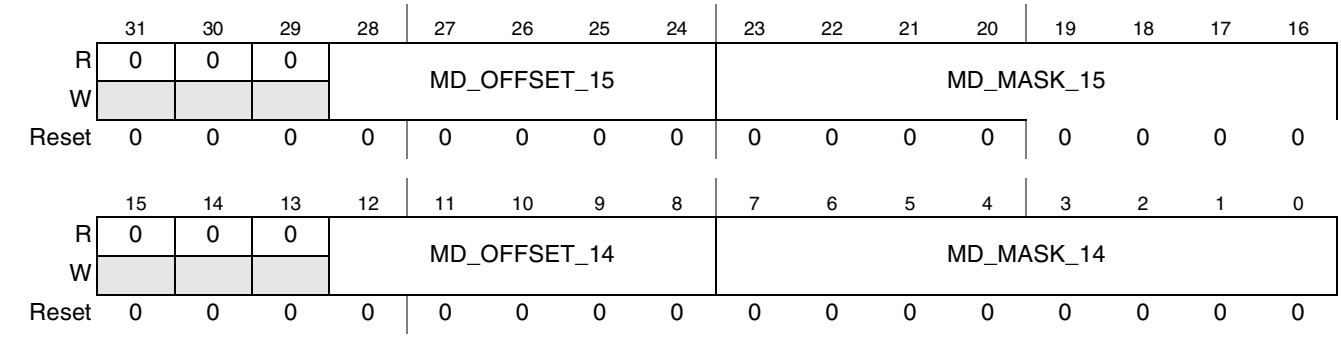


Figure 42-364. DC Mapping Configuration Register 22 (DC_MAP_CONF_22)

Table 42-367. DC_MAP_CONF_22 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_15	Mapping unit's offset parameter #15 This field defines the offset parameter #15 within the 24bit word coming from the DC.
23–16 MD_MASK_15	Mapping unit's mask value #15 This field defines the mask value #15 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_14	Mapping unit's offset parameter #14 This field defines the offset parameter #14 within the 24bit word coming from the DC.
7–0 MD_MASK_14	Mapping unit's mask value #14 This field defines the mask value #14 within the 8bit word coming from the DC.

42.2.3.12.90 DC Mapping Configuration Register 23 (DC_MAP_CONF_23)

Address 0xBASE+0xE058164 (DC_MAP_CONF_23) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_17					MD_MASK_17							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_16					MD_MASK_16							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-365. DC Mapping Configuration Register 23 (DC_MAP_CONF_23)

Table 42-368. DC_MAP_CONF_23 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_17	Mapping unit's offset parameter #17 This field defines the offset parameter #17 within the 24bit word coming from the DC.
23–16 MD_MASK_17	Mapping unit's mask value #17 This field defines the mask value #17 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_16	Mapping unit's offset parameter #16 This field defines the offset parameter #16 within the 24bit word coming from the DC.
7–0 MD_MASK_16	Mapping unit's mask value #16 This field defines the mask value #16 within the 8bit word coming from the DC.

42.2.3.12.91 DC Mapping Configuration Register 24 (DC_MAP_CONF_24)

Address 0xBASE+0xE058168 (DC_MAP_CONF_24) Access: User read/write

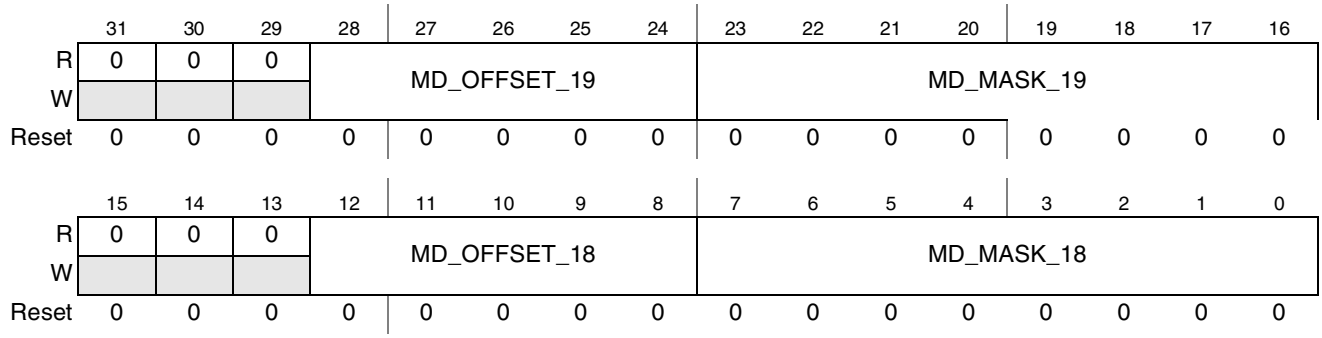


Figure 42-366. DC Mapping Configuration Register 24 (DC_MAP_CONF_24)

Table 42-369. DC_MAP_CONF_24 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_19	Mapping unit's offset parameter #19 This field defines the offset parameter #19 within the 24bit word coming from the DC.
23–16 MD_MASK_19	Mapping unit's mask value #19 This field defines the mask value #19 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_18	Mapping unit's offset parameter #18 This field defines the offset parameter #18 within the 24bit word coming from the DC.
7–0 MD_MASK_18	Mapping unit's mask value #18 This field defines the mask value #18 within the 8bit word coming from the DC.

42.2.3.12.92 DC Mapping Configuration Register 25 (DC_MAP_CONF_25)

Address 0xBASE+0xE05816C (DC_MAP_CONF_25) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_21					MD_MASK_21							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_20					MD_MASK_20							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-367. DC Mapping Configuration Register 25 (DC_MAP_CONF_25)

Table 42-370. DC_MAP_CONF_25 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_21	Mapping unit's offset parameter #21 This field defines the offset parameter #21 within the 24bit word coming from the DC.
23–16 MD_MASK_21	Mapping unit's mask value #21 This field defines the mask value #21 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_20	Mapping unit's offset parameter #20 This field defines the offset parameter #20 within the 24bit word coming from the DC.
7–0 MD_MASK_20	Mapping unit's mask value #20 This field defines the mask value #20 within the 8bit word coming from the DC.

42.2.3.12.93 DC Mapping Configuration Register 26 (DC_MAP_CONF_26)

Address 0xBASE+0xE058170 (DC_MAP_CONF_26) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	MD_OFFSET_23					MD_MASK_23							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	MD_OFFSET_22					MD_MASK_22							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-368. DC Mapping Configuration Register 26 (DC_MAP_CONF_26)

Table 42-371. DC_MAP_CONF_26 Field Descriptions

Field	Description
31–29	Reserved.
28–24 MD_OFFSET_23	Mapping unit's offset parameter #23 This field defines the offset parameter #23 within the 24bit word coming from the DC.
23–16 MD_MASK_23	Mapping unit's mask value #23 This field defines the mask value #23 within the 8bit word coming from the DC.
15–13	Reserved.
12–8 MD_OFFSET_22	Mapping unit's offset parameter #22 This field defines the offset parameter #22 within the 24bit word coming from the DC.
7–0 MD_MASK_22	Mapping unit's mask value #22 This field defines the mask value #22 within the 8bit word coming from the DC.

42.2.3.12.94 DC User General Data Event 0 Register 0 (DC_UGDE0_0)

Address 0xBASE+0xE058174 (DC_UGDE0_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	NF_NL_0		AUTO RESTART_0	ODD_EN_0	0	COD_ODD_START_0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COD_EV_START_0								0	COD_EV_PRIORITY_0			ID_CODED_0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-369. DC User General Data Event 0 Register 0 (DC_UGDE0_0)

Table 42-372. DC_UGDE0_0 Field Descriptions

Field	Description
31–29	Reserved.
28–27 NF_NL_0	the user may attach his general event #0 to New-line New-Frame and New-field events. One of these event triggers the user's general event #0's counter. The actual internal trigger is the pixel following the occurrence of the selected event. 00 - New Line 01 - New Frame 10 - New Field 11 - Reserved
26 AUTORESTART_0	User's general event #0 auto restart mode 0 - disable 1- User's general event #0's counter is automatically restarted.
25 ODD_EN_0	The user's general event #0 may be split into 2 internal signals. One one mode all the events are sent on one signal, on the second mode odd events are sent over one signal while even events are sent over the other signal (ODD_MODE) 1- ODD_MODE is enabled 0 - ODD_MODE is disabled
24	Reserved.
23–16 COD_ODD_START_0	This field holds a pointer in the microcode holding the routine to be performed following the user general event #0. When ODD_MODE is enabled, only the odd events will use this pointer When ODD_MODE is disabled this field is ignored
15–8 COD_EV_START_0	This field holds a pointer in the microcode holding the routine to be performed following the user general event #0. When ODD_MODE is enabled, only the even events will use this pointer When ODD_MODE is disabled, all the events will use this pointer
7	Reserved

Table 42-372. DC_UGDE0_0 Field Descriptions (continued)

Field	Description
6–3 COD_EV _PRIORI TY_0	This field defines the priority of the user general event #0 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
2–0 ID_COD ED_0	This field defines the number of DC channel number that user's general event #0 will be associated to. 000 - DC channel_0 001- DC channel_1 010 - DC channel_2 011 - DC channel_5 (DP_SYNC) 100 - DC channel_6 (DP ASYNC) 101:111 Reserved.

42.2.3.12.95 DC User General Data Event 0 Register 1 (DC_UGDE0_1)

Address 0xBASE+0xE058178 (DC_UGDE0_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-370. DC User General Data Event 0 Register 1 (DC_UGDE0_1)

Table 42-373. DC_UGDE0_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_0	This field holds the pre defined value that the counter counts too.

42.2.3.12.96 DC User General Data Event 0 Register 2 (DC_UGDE0_2)

Address 0xBASE+0xE05817C (DC_UGDE0_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OFFSET_DT_0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSET_DT_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-371. DC User General Data Event 0 Register 2 (DC_UGDE0_2)

Table 42-374. DC_UGDE0_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 OFFSET_DT_0	This field defines the offset value from which the counter of user general event #0 will start counting from

42.2.3.12.97 DC User General Data Event 0 Register 3 (DC_UGDE0_3)

Address 0xBASE+0xE058180 (DC_UGDE0_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_REPEAT_0												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_REPEAT_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-372. DC User General Data Event 0 Register 3 (DC_UGDE0_3)

Table 42-375. DC_UGDE0_3 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_REPEAT_0	When auto reload mode is disabled this field defines the number of events that will be generated by the user general event #0 mechanism

42.2.3.12.98 DC User General Data Event 1 Register 0 (DC_UGDE1_0)

Address 0xBASE+0xE058184 (DC_UGDE1_0) Access: User read/write

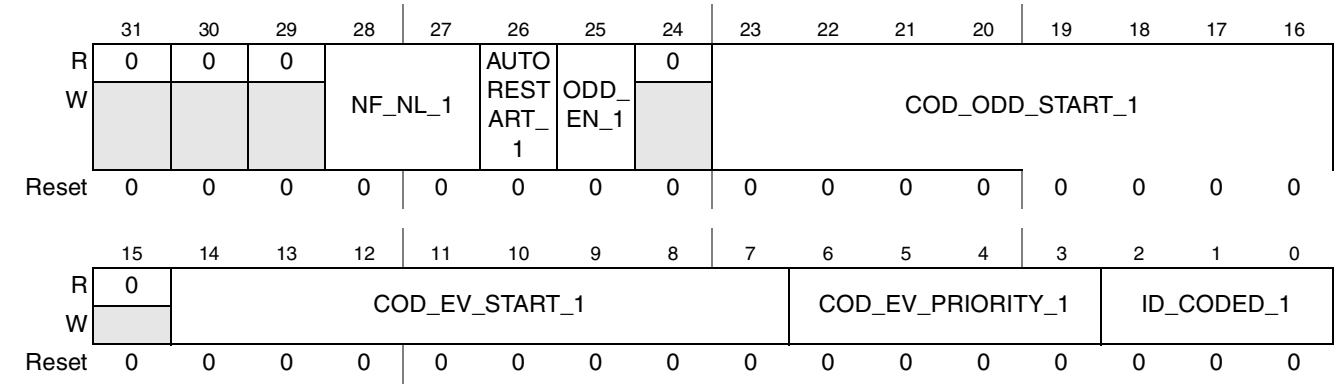


Figure 42-373. DC User General Data Event 1 Register 0 (DC_UGDE1_0)

Table 42-376. DC_UGDE1_0 Field Descriptions

Field	Description
31–29	Reserved.
28–27 NF_NL_1	the user may attach his general event #1 to New-line New-Frame and New-field events. One of these event triggers the user's general event #1's counter. The actual internal trigger is the pixel following the occurrence of the selected event. 00 - New Line 01 - New Frame 10 - New Field 11 - Reserved
26 AUTORESTART_1	User's general event #1 auto restart mode 0 - disable 1- User's general event #1's counter is automatically restarted.
25 ODD_EN_1	The user's general event #1 may be split into 2 internal signals. One one mode all the events are sent on one signal, on the second mode odd events are sent over one signal while even events are sent over the other signal (ODD_MODE) 1- ODD_MODE is enabled 0 - ODD_MODE is disabled
24	Reserved.
23–16 COD_ODD_START_1	This field holds a pointer in the microcode holding the routine to be performed following the user general event #1. When ODD_MODE is enabled, only the odd events will use this pointer When ODD_MODE is disabled this field is ignored
15–8 COD_EV_START_1	This field holds a pointer in the microcode holding the routine to be performed following the user general event #1. When ODD_MODE is enabled, only the even events will use this pointer When ODD_MODE is disabled, all the events will use this pointer

Table 42-376. DC_UGDE1_0 Field Descriptions (continued)

Field	Description
7	Reserved
6–3 COD_EV _PRIORI TY_1	This field defines the priority of the user general event #1 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
2–0 ID_COD ED_1	This field defines the number of DC channel number that user's general event #1 will be associated to 000 - DC channel_0 001- DC channel_1 010 - DC channel_2 011 - DC channel_5 (DP_SYNC) 100 - DC channel_6 (DP ASYNC) 101:111 Reserved.

42.2.3.12.99 DC User General Data Event 1 Register 1 (DC_UGDE1_1)

Address 0xBASE+0xE058188 (DC_UGDE1_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-374. DC User General Data Event 1 Register 1 (DC_UGDE1_1)

Table 42-377. DC_UGDE1_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_1	This field hold the pre defined value that the counter counts too

42.2.3.12.100DC User General Data Event 1 Register 2 (DC_UGDE1_2)

Address 0xBASE+0xE05818C (DC_UGDE1_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OFFSET_DT_1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSET_DT_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-375. DC User General Data Event 1 Register 2 (DC_UGDE1_2)

Table 42-378. DC_UGDE1_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 OFFSET_DT_1	This field defines the offset value from which the counter of user general event #1 will start counting from

42.2.3.12.101DC User General Data Event 1 Register 3 (DC_UGDE1_3)

Address 0xBASE+0xE058190 (DC_UGDE1_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_REPEAT_1												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_REPEAT_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-376. DC User General Data Event 1 Register 3 (DC_UGDE1_3)

Table 42-379. DC_UGDE1_3 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_REPEAT_1	When auto reload mode is disabled this field defines the number of events that will be generated by the user general event #1 mechanism

42.2.3.12.102DC User General Data Event 2 Register 0 (DC_UGDE2_0)

Address 0xBASE+0xE058194 (DC_UGDE2_0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	NF_NL_2		AUTO REST ART_ 2	ODD_ EN_2	0	COD_ODD_START_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	COD_EV_START_2							COD_EV_PRIORITY_2				ID_CODED_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-377. DC User General Data Event 2 Register 0 (DC_UGDE2_0)

Table 42-380. DC_UGDE2_0 Field Descriptions

Field	Description
31–29	Reserved.
28–27 NF_NL_ 2	the user may attach his general event #2 to New-line New-Frame and New-field events. One of these event triggers the user's general event #2's counter. The actual internal trigger is the pixel following the occurrence of the selected event. 00 - New Line 01 - New Frame 10 - New Field 11 - Reserved
26 AUTORE START_ 2	User's general event #2 auto restart mode 0 - disable 1- User's general event #2's counter is automatically restarted.
25 ODD_E N_2	The user's general event #2 may be split into 2 internal signals. One one mode all the events are sent on one signal, on the second mode odd events are sent over one signal while even events are sent over the other signal (ODD_MODE) 1- ODD_MODE is enabled 0 - ODD_MODE is disabled
24	Reserved.
23–16 COD_O DD_STA RT_2	This field holds a pointer in the microcode holding the routine to be performed following the user general event #2 When ODD_MODE is enabled, only the odd events will use this pointer When ODD_MODE is disabled this field is ignored
15–8 COD_EV _START _2	This field holds a pointer in the microcode holding the routine to be performed following the user general event #2. When ODD_MODE is enabled, only the even events will use this pointer When ODD_MODE is disabled, all the events will use this pointer

Table 42-380. DC_UGDE2_0 Field Descriptions (continued)

Field	Description
7	Reserved
6–3 COD_EV _PRIORI TY_2	This field defines the priority of the user general event #2 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
2–0 ID_COD ED_2	This field defines the number of DC channel number that user's general event #2 will be associated to 000 - DC channel_0 001- DC channel_1 010 - DC channel_2 011 - DC channel_5 (DP_SYNC) 100 - DC channel_6 (DP ASYNC) 101:111 Reserved.

42.2.3.12.103DC User General Data Event 2 Register 1 (DC_UGDE2_1)

Address 0xBASE+0xE058198 (DC_UGDE2_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_2												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-378. DC User General Data Event 2 Register 1 (DC_UGDE2_1)

Table 42-381. DC_UGDE2_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_2	This field hold the pre defined value that the counter counts too

42.2.3.12.104DC User General Data Event 2 Register 2 (DC_UGDE2_2)

Address 0xBASE+0xE05819C (DC_UGDE2_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OFFSET_DT_2												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSET_DT_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-379. DC User General Data Event 2 Register 2 (DC_UGDE2_2)

Table 42-382. DC_UGDE2_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 OFFSET_DT_2	This field defines the offset value from which the counter of user general event #2 will start counting from

42.2.3.12.105DC User General Data Event 2 Register 3 (DC_UGDE2_3)

Address 0xBASE+0xE0581A0 (DC_UGDE2_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_REPEAT_2												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_REPEAT_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-380. DC User General Data Event 2 Register 3 (DC_UGDE2_3)

Table 42-383. DC_UGDE2_3 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_REPEAT_2	When auto reload mode is disabled this field defines the number of events that will be generated by the user general event #2 mechanism

42.2.3.12.106DC User General Data Event 1 Register 0 (DC_UGDE3_0)

Address 0xBASE+0xE0581A4 (DC_UGDE3_0) Access: User read/write

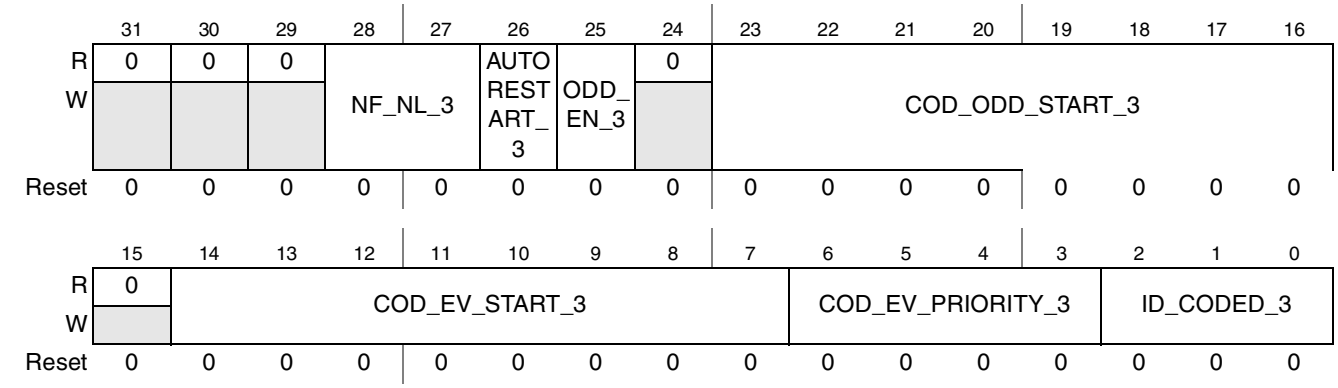


Figure 42-381. DC User General Data Event 1 Register 0 (DC_UGDE3_0)

Table 42-384. DC_UGDE3_0 Field Descriptions

Field	Description
31–29	Reserved.
28–27 NF_NL_3	the user may attach his general event #3 to New-line New-Frame and New-field events. One of these event triggers the user's general event #3's counter. The actual internal trigger is the pixel following the occurrence of the selected event. 00 - New Line 01 - New Frame 10 - New Field 11 - Reserved
26 AUTORESTART_3	User's general event #3 auto restart mode 0 - disable 1- User's general event #3's counter is automatically restarted.
25 ODD_EN_3	The user's general event #3 may be split into 2 internal signals. One one mode all the events are sent on one signal, on the second mode odd events are sent over one signal while even events are sent over the other signal (ODD_MODE) 1- ODD_MODE is enabled 0 - ODD_MODE is disabled
24	Reserved.
23–16 COD_ODD_START_3	This field holds a pointer in the microcode holding the routine to be performed following the user general event #3. When ODD_MODE is enabled, only the odd events will use this pointer When ODD_MODE is disabled this field is ignored
15–8 COD_EV_START_3	This field holds a pointer in the microcode holding the routine to be performed following the user general event #3. When ODD_MODE is enabled, only the even events will use this pointer When ODD_MODE is disabled, all the events will use this pointer

Table 42-384. DC_UGDE3_0 Field Descriptions (continued)

Field	Description
7	Reserved
6–3 COD_EV _PRIORI TY_3	This field defines the priority of the user general event #3 0000 - disable 0001 - Priority #1 (lowest) 0010 - Priority #2 ... 1101 - Priority #13 (highest) 1110 - Reserved 1111 - Reserved The priority between the events should be set to a unique value. i.e. two events must not have the same priority (except 0000 - disable)
2–0 ID_COD ED_3	This field defines the number of DC channel number that user's general event #3 will be associated to 000 - DC channel_0 001- DC channel_1 010 - DC channel_2 011 - DC channel_5 (DP_SYNC) 100 - DC channel_6 (DP ASYNC) 101:111 Reserved.

42.2.3.12.107DC User General Data Event 3 Register 1 (DC_UGDE3_1)

Address 0xBASE+0xE0581A8 (DC_UGDE3_1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_3												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-382. DC User General Data Event 3 Register 1 (DC_UGDE3_1)

Table 42-385. DC_UGDE3_1 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_3	This field hold the pre defined value that the counter counts too

42.2.3.12.108DC User General Data Event 3 Register 2 (DC_UGDE3_2)

Address 0xBASE+0xE0581AC (DC_UGDE3_2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OFFSET_DT_3												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSET_DT_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-383. DC User General Data Event 3 Register 2 (DC_UGDE3_2)

Table 42-386. DC_UGDE3_2 Field Descriptions

Field	Description
31–29	Reserved.
28–0 OFFSET_DT_3	This field defines the offset value from which the counter of user general event #3 will start counting from

42.2.3.12.109DC User General Data Event 3 Register 3 (DC_UGDE3_3)

Address 0xBASE+0xE0581B0 (DC_UGDE3_3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	STEP_REPEAT_3												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STEP_REPEAT_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-384. DC User General Data Event 3 Register 3 (DC_UGDE3_3)

Table 42-387. DC_UGDE3_3 Field Descriptions

Field	Description
31–29	Reserved.
28–0 STEP_REPEAT_3	When auto reload mode is disabled this field defines the number of events that will be generated by the user general event #3 mechanism

42.2.3.12.110DC Low Level Access Control Register 0 (DC_LLA0)

Address 0xBASE+0xE0581B4 (DC_LLA0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCU_RS_3_0								MCU_RS_2_0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCU_RS_1_0								MCU_RS_0_0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-385. DC Low Level Access Control Register 0 (DC_LLA0)

Table 42-388. DC_LLA0 Field Descriptions

Field	Description
31–24 MCU_RS_3_0	This field holds a pointer in the microcode handling the RS_3 routine for the display defined at DISP_ID_8, when in Low level access mode,
23–16 MCU_RS_2_0	This field holds a pointer in the microcode handling the RS_2 routine for the display defined at DISP_ID_8, when in Low level access mode,
15–8 MCU_RS_1_0	This field holds a pointer in the microcode handling the RS_1 routine for the display defined at DISP_ID_8, when in Low level access mode,
7–0 MCU_RS_0_0	This field holds a pointer in the microcode handling the RS_0 routine for the display defined at DISP_ID_8, when in Low level access mode,

42.2.3.12.111DC Low Level Access Control Register 1 (DC_LLA1)

Address 0xBASE+0xE0581B8 (DC_LLA1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCU_RS_3_1								MCU_RS_2_1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCU_RS_1_1								MCU_RS_0_1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-386. DC Low Level Access Control Register 1 (DC_LLA1)

Table 42-389. DC_LLA1 Field Descriptions

Field	Description
31–24 MCU_RS_3_1	This field holds a pointer in the microcode handling the RS_3 routine for the display defined at DISP_ID_9, when in Low level access mode,
23–16 MCU_RS_2_1	This field holds a pointer in the microcode handling the RS_2 routine for the display defined at DISP_ID_9, when in Low level access mode,
15–8 MCU_RS_1_1	This field holds a pointer in the microcode handling the RS_1 routine for the display defined at DISP_ID_9, when in Low level access mode,
7–0 MCU_RS_0_1	This field holds a pointer in the microcode handling the RS_0 routine for the display defined at DISP_ID_9, when in Low level access mode,

42.2.3.12.112DC Low Level Read Access Control Register 0 (DC_R_LLA0)

Address 0xBASE+0xE0581BC (DC_R_LLA0) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCU_RS_R_3_0								MCU_RS_R_2_0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCU_RS_R_1_0								MCU_RS_R_0_0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-387. DC Low Level Read Access Control Register 0 (DC_R_LLA0)

Table 42-390. DC_R_LLA0 Field Descriptions

Field	Description
31–24 MCU_RS_3_0	This field holds a pointer in the microcode handling the RS_3 routine for the display defined at DISP_ID_8, when in Read Low level access mode,
23–16 MCU_RS_2_0	This field holds a pointer in the microcode handling the RS_2 routine for the display defined at DISP_ID_8, when in Read Low level access mode,
15–8 MCU_RS_R_1_0	This field holds a pointer in the microcode handling the RS_1 routine for the display defined at DISP_ID_8, when in Read Low level access mode,
7–0 MCU_RS_R_0_0	This field holds a pointer in the microcode handling the RS_0 routine for the display defined at DISP_ID_8, when in Read Low level access mode,

42.2.3.12.113DC Read Low Level Access Control Register 1 (DC_R_LLA1)

Address 0xBASE+0xE0581C0 (DC_R_LLA1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MCU_RS_R_3_1								MCU_RS_R_2_1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MCU_RS_R_1_1								MCU_RS_R_0_1							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-388. DC Read Low Level Access Control Register 1 (DC_R_LLA1)

Table 42-391. DC_R_LLA1 Field Descriptions

Field	Description
31–24 MCU_RS_R_3_1	This field holds a pointer in the microcode handling the RS_3 routine for the display defined at DISP_ID_9, when in Read Low level access mode,
23–16 MCU_RS_R_2_1	This field holds a pointer in the microcode handling the RS_2 routine for the display defined at DISP_ID_9, when in Read Low level access mode,

Table 42-391. DC_R_LLA1 Field Descriptions (continued)

Field	Description
15–8 MCU_RS_R_1_1	This field holds a pointer in the microcode handling the RS_1 routine for the display defined at DISP_ID_9, when in Read Low level access mode,
7–0 MCU_RS_R_0_1	This field holds a pointer in the microcode handling the RS_0 routine for the display defined at DISP_ID_9, when in Read Low level access mode,

42.2.3.12.114DC Write Channel 5 Configuration Register (DC_WR_CH_ADDR_5_ALT)

Address 0xBASE+0xE0581C4 (DC_WR_CH_ADDR_5_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	ST_ADDR_5_ALT												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ST_ADDR_5_ALT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-389. DC Write Channel 5 Configuration Register (DC_WR_CH_ADDR_5_ALT)

Table 42-392. DC_WR_CH_ADDR_5_ALT Field Descriptions

Field	Description
31–29	Reserved.
28–0 ST_ADDR_5_ALT	This field defines the start address within the display's memory space where the write transactions will be done to for channel #5, when alternate flow is performed via channel #5

42.2.3.12.115DC Status Register (DC_STAT)

Address 0xBASE+0xE0581C8 (DC_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DC_TRIPLE_BUF_DATA_EMPTY_1	DC_TRIPLE_BUF_DATA_FULL_1	DC_TRIPLE_BUF_CNT_EMPTY_1	DC_TRIPLE_BUF_CNT_FULL_1	DC_TRIPLE_BUF_DATA_EMPTY_0	DC_TRIPLE_BUF_DATA_FULL_0	DC_TRIPLE_BUF_CNT_EMPTY_0	DC_TRIPLE_BUF_CNT_FULL_0
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0

Figure 42-390. DC Status Register (DC_STAT)

Table 42-393. DC_STAT Field Descriptions

Field	Description
31–8	Reserved.
7 DC_TRIPLE_BUF_DATA_EMPTY_1	
6 DC_TRIPLE_BUF_DATA_FULL_1	
5 DC_TRIPLE_BUF_CNT_EMPTY_1	

Table 42-393. DC_STAT Field Descriptions (continued)

Field	Description
4 DC_TRIPLE_BUF_CNT_FULL_1	
3 DC_TRIPLE_BUF_DATA_EMPTY_0	
2 DC_TRIPLE_BUF_DATA_FULL_0	
1 DC_TRIPLE_BUF_CNT_EMPTY_0	
0 DC_TRIPLE_BUF_CNT_FULL_0	

42.2.3.13 DMFC Registers

42.2.3.13.1 DMFC Read Channel Register (DMFC_RD_CHAN)

Address 0xBASE+0xE060000 (DMFC_RD_CHAN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	dmfc_ppw_c		dmfc_wm_clr_0			dmfc_wm_set_0			dmfc_wm_en_0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	dmfc_burst_size_0		0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-391. DMFC Read Channel Register (DMFC_RD_CHAN)

Table 42-394. DMFC_RD_CHAN Field Descriptions

Field	Description
31–26	Reserved.
25–24 dmfc_ppw_c	Pixel Per Word coded. This field defines the size of the read data from the display. 00 8 bit per pixel 01 16 bit per pixel 10 24 (rgb) bit per pixel or 32 bit per pixel 11 - Reserved
23-21 dmfc_wm_clr_0	Watermark Clear This field defines the watermark's level of the DMFC read FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of free bursts at the FIFO (dmfc_wm_clr_0 > dmfc_wm_set_0)
20-18 dmfc_wm_set_0	Watermark Set This field defines the watermark's level of the DMFC read FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of free bursts at the FIFO (dmfc_wm_clr_0 > dmfc_wm_set_0)
17 dmfc_wm_en_0	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
16–8	Reserved.
7-6 dmfc_burst_size_0	Read burst Size This field defines the burst size of the DMFC's read accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, going to the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
5–0	Reserved.

42.2.3.13.2 DMFC Write Channel Register (DMFC_WR_CHAN)

Address 0xBASE+0xE060004 (DMFC_WR_CHAN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	dmfc_burst_size_2c		dmfc_fifo_size_2c			dmfc_st_addr_2c			dmfc_burst_size_1c		dmfc_fifo_size_1c			dmfc_st_addr_1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	dmfc_burst_size_2		dmfc_fifo_size_2			dmfc_st_addr_2			dmfc_burst_size_1		dmfc_fifo_size_1			dmfc_st_addr_1		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-392. DMFC Write Channel Register (DMFC_WR_CHAN)

Table 42-395. DMFC_WR_CHAN Field Descriptions

Field	Description
31–30 dmfc_burst_size_2c	Burst size of IDMAC's channel 43 This field defines the burst size of the IDMAC's channel 43 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
29-27 dmfc_fifo_size_2c	DMFC FIFO size for IDMAC's channel 43 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 43 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
26–24 dmfc_st_addr_2c	DMFC Start Address for IDMAC's channel 43 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 43. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
23–22 dmfc_burst_size_1c	Burst size of IDMAC's channel 42 This field defines the burst size of the IDMAC's channel 42 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
21-19 dmfc_fifo_size_1c	DMFC FIFO size for IDMAC's channel 42 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 42 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel

Table 42-395. DMFC_WR_CHAN Field Descriptions (continued)

Field	Description
18–16 dmfc_st_addr_1c	DMFC Start Address for IDMAC's channel 42 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 42. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
15–14 dmfc_burst_size_2	Burst size of IDMAC's channel 41 This field defines the burst size of the IDMAC's channel 41 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
13-11 dmfc_fifo_size_2	DMFC FIFO size for IDMAC's channel 41 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 41 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
10–8 dmfc_st_addr_2	DMFC Start Address for IDMAC's channel 41 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 41. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
7–6 dmfc_burst_size_1	Burst size of IDMAC's channel 28 This field defines the burst size of the IDMAC's channel 28 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit

Table 42-395. DMFC_WR_CHAN Field Descriptions (continued)

Field	Description
5-3 dmfc_fifo_size_1	DMFC FIFO size for IDMAC's channel 28 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 28 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
2–0 dmfc_st_addr_1	DMFC Start Address for IDMAC's channel 28 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 28. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7

42.2.3.13.3 DMFC Write Channel Definition Register (DMFC_WR_CHAN_DEF)

Address 0xBASE+0xE060008 (DMFC_WR_CHAN_DEF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	dmfc_wm_clr_2c			dmfc_wm_set_2c			dmfc_wm_en_2c	0	dmfc_wm_clr_1c			dmfc_wm_set_1c			dmfc_wm_en_1c	0
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	dmfc_wm_clr_2			dmfc_wm_set_2			dmfc_wm_en_2	0	dmfc_wm_clr_1			dmfc_wm_set_1			dmfc_wm_en_1	0
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 42-393. DMFC Write Channel Definition Register (DMFC_WR_CHAN_DEF)

Table 42-396. DMFC_WR_CHAN_DEF Field Descriptions

Field	Description
31-29 dmfc_wm_clr_2c	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
28-26 dmfc_wm_set_2c	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
25 dmfc_wm_en_2c	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
24	Reserved
23-21 dmfc_wm_clr_1c	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
20-18 dmfc_wm_set_1c	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
17 dmfc_wm_en_1c	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
16	Reserved
15-13 dmfc_wm_clr_2	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
12-10 dmfc_wm_set_2	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
9 dmfc_wm_en_2	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
8	Reserved
7-5 dmfc_wm_clr_1	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
4-2 dmfc_wm_set_1	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set

Table 42-396. DMFC_WR_CHAN_DEF Field Descriptions (continued)

Field	Description
1 dmfc_wm_en_1	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
0	Reserved

42.2.3.13.4 DMFC Display Processor Channel Register (DMFC_DP_CHAN)

Address 0xBASE+0xE06000C (DMFC_DP_CHAN) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	dmfc_burst_size_6f		dmfc_fifo_size_6f			dmfc_st_addr_6f			dmfc_burst_size_6b		dmfc_fifo_size_6b			dmfc_st_addr_6b		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	dmfc_burst_size_5f		dmfc_fifo_size_5f			dmfc_st_addr_5f			dmfc_burst_size_5b		dmfc_fifo_size_5b			dmfc_st_addr_5b		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-394. DMFC Display Processor Channel Register (DMFC_DP_CHAN)

Table 42-397. DMFC_DP_CHAN Field Descriptions

Field	Description
31–16	Reserved.
31–30 dmfc_burst_size_6f	Burst size of IDMAC's channel 29 This field defines the burst size of the IDMAC's channel 29 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
29-27 dmfc_fifo_size_6f	DMFC FIFO size for IDMAC's channel 29 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 29 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel

Table 42-397. DMFC_DP_CHAN Field Descriptions (continued)

Field	Description
26–24 dmfc_st_addr_6f	DMFC Start Address for IDMAC's channel 29 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 29. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
23–22 dmfc_burst_size_6b	Burst size of IDMAC's channel 24 This field defines the burst size of the IDMAC's channel 24 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
21-19 dmfc_fifo_size_6b	DMFC FIFO size for IDMAC's channel 24 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 24 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
18–16 dmfc_st_addr_6b	DMFC Start Address for IDMAC's channel 24 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 24. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
15–14 dmfc_burst_size_5f	Burst size of IDMAC's channel 27 This field defines the burst size of the IDMAC's channel 27 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit

Table 42-397. DMFC_DP_CHAN Field Descriptions (continued)

Field	Description
13-11 dmfc_fifo_size_5f	DMFC FIFO size for IDMAC's channel 27 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 27 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
10–8 dmfc_st_addr_5f	DMFC Start Address for IDMAC's channel 27 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 27. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
7–6 dmfc_burst_size_5b	Burst size of IDMAC's channel 23 This field defines the burst size of the IDMAC's channel 23 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
5-3 dmfc_fifo_size_5b	DMFC FIFO size for IDMAC's channel 23 This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 23 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
2–0 dmfc_st_addr_5b	DMFC Start Address for IDMAC's channel 23 This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 23. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7

42.2.3.13.5 DMFC Display Channel Definition Register (DMFC_DP_CHAN_DEF)

Address 0xBASE+0xE060010 (DMFC_DP_CHAN_DEF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	dmfc_wm_clr_6f			dmfc_wm_set_6f			dmfc_wm_en_6f	0	dmfc_wm_clr_6b			dmfc_wm_set_6b			dmfc_wm_en_6b	0
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dmfc_wm_clr_5f			dmfc_wm_set_5f			dmfc_wm_en_5f	0	dmfc_wm_clr_5b			dmfc_wm_set_5b			dmfc_wm_en_5b	0
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 42-395. DMFC Display Channel Definition Register (DMFC_DP_CHAN_DEF)

Table 42-398. DMFC_DP_CHAN_DEF Field Descriptions

Field	Description
31-29 dmfc_wm_clr_6f	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
28-26 dmfc_wm_set_6f	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
25 dmfc_wm_en_6f	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
24	Reserved
23-21 dmfc_wm_clr_6b	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
20-18 dmfc_wm_set_6b	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
17 dmfc_wm_en_6b	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
16	Reserved
15-13 dmfc_wm_clr_5f	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)

Table 42-398. DMFC_DP_CHAN_DEF Field Descriptions (continued)

Field	Description
12-10 dmfc_wm_set_5f	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
9 dmfc_wm_en_5f	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
8	Reserved
7-5 dmfc_wm_clr_5b	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
4-2 dmfc_wm_set_5b	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
1 dmfc_wm_en_5b	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
0	Reserved

42.2.3.13.6 DMFC General 1 Register (DMFC_GENERAL1)

Address 0xBASE+0xE060014 (DMFC_GENERAL1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	WAIT4EOT_9	WAIT4EOT_6F	WAIT4EOT_6B	WAIT4EOT_5F	WAIT4EOT_5B	WAIT4EOT_4	WAIT4EOT_3	WAIT4EOT_2	WAIT4EOT_1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dmfc_wm_clr_9			dmfc_wm_set_9			dmfc_wm_en_9	0	0	dmfc_burst_size_9		0	0	0	dmfc_dcdp_sync_pr	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 42-396. DMFC General 1 Register (DMFC_GENERAL1)

Table 42-399. DMFC_GENERAL1 Field Descriptions

Field	Description
31–27	Reserved.
23 WAIT4EOT_9	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #9 is in wait4eot mode 0 - FIFO #9 is in normal mode
23 WAIT4EOT_6F	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #6F is in wait4eot mode 0 - FIFO #6F is in normal mode
22 WAIT4EOT_6B	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #6B is in wait4eot mode 0 - FIFO #6B is in normal mode
21 WAIT4EOT_5F	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #5F is in wait4eot mode 0 - FIFO #5F is in normal mode
20 WAIT4EOT_5B	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #5B is in wait4eot mode 0 - FIFO #5B is in normal mode
19 WAIT4EOT_4	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #4 is in wait4eot mode 0 - FIFO #4 is in normal mode
18 WAIT4EOT_3	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #3 is in wait4eot mode 0 - FIFO #3 is in normal mode
17 WAIT4EOT_2	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #2 is in wait4eot mode 0 - FIFO #2 is in normal mode
16 WAIT4EOT_1	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #1 is in wait4eot mode 0 - FIFO #1 is in normal mode

Table 42-399. DMFC_GENERAL1 Field Descriptions (continued)

Field	Description
15-13 dmfc_wm_clr_9	Watermark Clear This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
12-10 dmfc_wm_set_9	Watermark Set This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
9 dmfc_wm_en_9	Watermark enable. This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
8	Reserved
6-5 dmfc_burst_size_9	Burst size of IDMAC's channel 44 This field defines the burst size of the IDMAC's channel 44 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. This channel is targeted for MASK - the FIFO size is always 32X128; The base address is always the upper half of the 8th segment 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
4-2	Reserved.
1–0 dmfc_dcdp_sync_pr	DMFC's memory access priority settings for simultaneous synchronous flows from DC & DP 00 - Forbidden - should not be used. 01 - DC has higher priority over DP 10 - DP has higher priority over DC 11 - Round Robin

42.2.3.13.7 DMFC General Register 2 (DMFC_GENERAL2)

Address 0xBASE+0xE060018 (DMFC_GENERAL2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	dmfc_frame_height_rd												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	dmfc_frame_width_rd												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-397. DMFC General Register 2 (DMFC_GENERAL2)

Table 42-400. DMFC_GENERAL2 Field Descriptions

Field	Description
31–29	Reserved.
28–16 dmfc_frame_height_rd	Frame height for read channel from the display to the IDMAC; Units are pixels
15–13	Reserved.
12–0 dmfc_frame_width_rd	Frame width for read channel from the display to the IDMAC; Units are pixels

42.2.3.13.8 DMFC IC Interface Control Register (DMFC_IC_CTRL)

Address 0xBASE+0xE06001C (DMFC_IC_CTRL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	dmfc_ic_frame_height_rd													dmfc_ic_frame_width_rd		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	dmfc_ic_frame_width_rd										dmfc_ic_ppw_c		0	dmfc_ic_in_port		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 42-398. DMFC General 1 Register (DMFC_GENERAL1)

Table 42-401. DMFC_GENERAL Field Descriptions

Field	Description
31–19 dmfc_ic_frame_height_rd	Frame's height for the channel coming from IC. Units are lines
18–6 dmfc_ic_frame_width_rd	Frame's width for the channel coming from IC. Units are pixels
5–4 dmfc_ic_ppw_c	Pixel Per Word coded from IC. This field defines the size of the data coming from the IC. 00 8 bit per pixel 01 16 bit per pixel 10 24 bit per pixel 11 - Reserved

Table 42-401. DMFC_GENERAL Field Descriptions (continued)

Field	Description
3	Reserved
2–0 dmfc_ic_in_port	DMFC input port When data is coming from the IC, the IC channel replaces one of the IDMAC's channels connected to the DMFC. This field defines which IDMAC's channel is replaced by the IC channel. 000 - CH28 001 - CH41 010 - Reserved, IC channel is disabled 011 - Reserved, IC channel is disabled 100 - CH23 101 - CH27 110 - CH24 111 - CH29

42.2.3.13.9 DMFC Write Channel Alternate Register (DMFC_WR_CHAN_ALT)

Address 0xBASE+0xE060020 (DMFC_WR_CHAN_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dmfc_burst_size_2_alt		dmfc_fifo_size_2_alt			dmfc_st_addr_2_alt			0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-399. DMFC Write Channel Alternate Register (DMFC_WR_CHAN_ALT)

Table 42-402. DMFC_WR_CHAN_ALT Field Descriptions

Field	Description
31–16	Reserved
15–14 dmfc_burst_size_2_alt	Burst size of IDMAC's channel 41 (for alternate flow) This field defines the burst size of the IDMAC's channel 41 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit

Table 42-402. DMFC_WR_CHAN_ALT Field Descriptions (continued)

Field	Description
13-11 dmfc_fifo_size_2_alt	DMFC FIFO size for IDMAC's channel 41 (for alternate flow) This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 41 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
10–8 dmfc_st_addr_2_alt	DMFC Start Address for IDMAC's channel 41 (for alternate flow) This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 41. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
7–0	Reserved

42.2.3.13.10 DMFC Write Channel Definition Alternate Register (DMFC_WR_CHAN_DEF_ALT)

Address 0xBASE+0xE060024 (DMFC_WR_CHAN_DEF_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dmfc_wm_clr_2_alt			dmfc_wm_set_2_alt			dmfc_wm_en_2_alt	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-400. DMFC Write Channel Definition Alternate Register (DMFC_WR_CHAN_DEF_ALT)

Table 42-403. DMFC_WR_CHAN_DEF_ALT Field Descriptions

Field	Description
31-16	Reserved
15-13 dmfc_wm_clr_2_alt	Watermark Clear (for alternate flow) This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
12-10 dmfc_wm_set_2_alt	Watermark Set (for alternate flow) This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
9 dmfc_wm_en_2_alt	Watermark enable. (for alternate flow) This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
7-0	Reserved

42.2.3.13.11 DMFC Display Processor Channel Alternate Register (DMFC_DP_CHAN_ALT)

Address 0xBASE+0xE060028 (DMFC_DP_CHAN_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	dmfc_burst_size_6f_alt		dmfc_fifo_size_6f_alt			dmfc_st_addr_6f_alt			dmfc_burst_size_6b_alt		dmfc_fifo_size_6b_alt			dmfc_st_addr_6b_alt		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	dmfc_burst_size_5b_alt		dmfc_fifo_size_5b_alt			dmfc_st_addr_5b_alt		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-401. DMFC Display Processor Channel Alternate Register (DMFC_DP_CHAN_ALT)

Table 42-404. DMFC_DP_CHAN_ALT Field Descriptions

Field	Description
31–16	Reserved.
31–30 dmfc_burst_size_6f_alt	Burst size of IDMAC's channel 29 (for alternate flow) This field defines the burst size of the IDMAC's channel 29 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit

Table 42-404. DMFC_DP_CHAN_ALT Field Descriptions (continued)

Field	Description
29-27 dmfc_fifo_size_6f_alt	DMFC FIFO size for IDMAC's channel 29 (for alternate flow) This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 29 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
26–24 dmfc_st_addr_6f_alt	DMFC Start Address for IDMAC's channel 29 (for alternate flow) This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 29. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
23–22 dmfc_burst_size_6b_alt	Burst size of IDMAC's channel 24 (for alternate flow) This field defines the burst size of the IDMAC's channel 24 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
21-19 dmfc_fifo_size_6b_alt	DMFC FIFO size for IDMAC's channel 24 (for alternate flow) This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 24 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
18–16 dmfc_st_addr_6b_alt	DMFC Start Address for IDMAC's channel 24 (for alternate flow) This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 24. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7
15–8	Reserved

Table 42-404. DMFC_DP_CHAN_ALT Field Descriptions (continued)

Field	Description
7–6 dmfc_burst_size_5b_alt	Burst size of IDMAC's channel 23 (for alternate flow) This field defines the burst size of the IDMAC's channel 23 write accesses. This settings must match the settings in the corresponding IDMAC channel's settings. 00 - 32 words of 128 bit (4 pixels of 32 bit each, coming from the IDMAC) 01 - 16 words of 128 bit 10 - 8 words of 128 bit 11 - 4 words of 128 bit
5-3 dmfc_fifo_size_5b_alt	DMFC FIFO size for IDMAC's channel 23 (for alternate flow) This field defines the FIFO partition for the DMFC channel connected to IDMAC's channel 23 000 - All (512X128 words) the DMFC's FIFO is allocated to this channel 001 - 256X128 words are allocated to this channel 010 - 128X128 words are allocated to this channel 011 - 64X128 words are allocated to this channel 100 - 32X128 words are allocated to this channel 101- 16X128 words are allocated to this channel 110 - 8X128 words are allocated to this channel 111 - 4X128 words are allocated to this channel
2–0 dmfc_st_addr_5b_alt	DMFC Start Address for IDMAC's channel 23 (for alternate flow) This field defines the base address at the DMFC's FIFO of the partition allocated to the channel connected to IDMAC's channel 23. The FIFO is partitioned to 8 equal segments. Each segment is 64X128 words The value of this field is the number of the segment 000 - Segment 0 001 - Segment 1 ... 111 - Segment 7

42.2.3.13.12 DMFC Display Channel Definition Alternate Register (DMFC_DP_CHAN_DEF_ALT)

Address 0xBASE+0xE06002C (DMFC_DP_CHAN_DEF_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	dmfc_wm_clr_6f_alt				dmfc_wm_set_6f_alt			0	dmfc_wm_clr_6b_alt				dmfc_wm_set_6b_alt			0
W							dmfc_wm_en_6f_alt								dmfc_wm_en_6b_alt	
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	dmfc_wm_clr_5b_alt				dmfc_wm_set_5b_alt			0
W															dmfc_wm_en_5b_alt	
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 42-402. DMFC Display Channel Definition Register (DMFC_DP_CHAN_DEF)

Table 42-405. DMFC_DP_CHAN_DEF_ALT Field Descriptions

Field	Description
31-29 dmfc_wm_clr_6f_alt	Watermark Clear (for alternate flow) This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
28-26 dmfc_wm_set_6f_alt	Watermark Set (for alternate flow) This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
25 dmfc_wm_en_6f_alt	Watermark enable. (for alternate flow) This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
24	Reserved
23-21 dmfc_wm_clr_6b_alt	Watermark Clear (for alternate flow) This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
20-18 dmfc_wm_set_6b_alt	Watermark Set (for alternate flow) This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
17 dmfc_wm_en_6b_alt	Watermark enable. (for alternate flow) This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
16-8	Reserved
7-5 dmfc_wm_clr_5b_alt	Watermark Clear (for alternate flow) This field defines the watermark's level of the DMFC's write FIFO. Crossing this level will clear the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set)
4-2 dmfc_wm_set_5b_alt	Watermark Set (for alternate flow) This field defines the watermark's level of the DMFC write FIFO. Crossing this level will send the watermark signal to the IDMAC. The WM level is the amount of occupied bursts at the FIFO (dmfc_wm_clr > dmfc_wm_set
1 dmfc_wm_en_5b_alt	Watermark enable. (for alternate flow) This bit enables the watermark feature of the FIFO 1 - WM feature is enabled 0 - WM feature is disabled
0	Reserved

42.2.3.13.13 DMFC General 1 Altenate Register (DMFC_GENERAL1_ALT)

Address 0xBASE+0xE060030 (DMFC_GENERAL1_ALT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	WAIT4EOT_6F_ALT	WAIT4EOT_6B_ALT	0	WAIT4EOT_5B_ALT	0	0	WAIT4EOT_2_ALT	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-403. DMFC General 1 Altenate Register (DMFC_GENERAL1_ALT)

Table 42-406. DMFC_GENERAL1_ALT Field Descriptions

Field	Description
31–24	Reserved.
23 WAIT4EOT_6F_ALT	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #6F is in wait4eot mode (for alternate flow) 0 - FIFO #6F is in normal mode (for alternate flow)
22 WAIT4EOT_6B_ALT	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #6B is in wait4eot mode (for alternate flow) 0 - FIFO #6B is in normal mode (for alternate flow)
21	Reserved
20 WAIT4EOT_5B_ALT	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #5B is in wait4eot mode (for alternate flow) 0 - FIFO #5B is in normal mode (for alternate flow)
19-18	Reserved
17 WAIT4EOT_2_ALT	In normal operation the DMFC sends requests to the IDMAC whenever there is room in the FIFO. When this bit is set the DMFC sends the request only after the current transfer is terminated, (wait4eot mode) When the FIFO is larger then the size of the line, the user should work in wait4eot mode. 1 - FIFO #2 is in wait4eot mode (for alternate flow) 0 - FIFO #2 is in normal mode (for alternate flow)
16-0	Reserved.

42.2.3.13.14 DMFC Status Register (DMFC_STAT)

This register contains DMFC's status bits. All the bits in this register are read-only.

Address 0xBASE+0xE060034 (DMFC_STAT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	DMFC_IC_BUFFER_EMPTY	DMFC_IC_BUFFER_FULL	DMFC_FIFO_EMPTY_11	DMFC_FIFO_EMPTY_10	DMFC_FIFO_EMPTY_9	DMFC_FIFO_EMPTY_8	DMFC_FIFO_EMPTY_7	DMFC_FIFO_EMPTY_6	DMFC_FIFO_EMPTY_5	DMFC_FIFO_EMPTY_4
W																
Reset	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMFC_FIFO_EMPTY_3	DMFC_FIFO_EMPTY_2	DMFC_FIFO_EMPTY_1	DMFC_FIFO_EMPTY_0	DMFC_FIFO_FULL_11	DMFC_FIFO_FULL_10	DMFC_FIFO_FULL_9	DMFC_FIFO_FULL_8	DMFC_FIFO_FULL_7	DMFC_FIFO_FULL_6	DMFC_FIFO_FULL_5	DMFC_FIFO_FULL_4	DMFC_FIFO_FULL_3	DMFC_FIFO_FULL_2	DMFC_FIFO_FULL_1	DMFC_FIFO_FULL_0
W																
Reset	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-404. DMFC Status Register (DMFC_STAT)

Table 42-407. DMFC_STAT Field Descriptions

Field	Description
31-26	Reserved
25 DMFC_IC_BUFFER_EMPTY	This bit indicates on a IC FIFO, inside the DMFC, empty condition. 0 IC FIFO not empty 1 IC FIFO is empty

Table 42-407. DMFC_STAT Field Descriptions (continued)

Field	Description
24 DMFC_IC_BUFFER_FULL	This bit indicates on a IC FIFO, inside the DMFC, full condition. 0 IC FIFO not full 1 IC FIFO is full
23-12 DMFC_FIFO_EMPTY_<i>	This bit indicates on a DMFC FIFO#<i> empty condition. 0 FIFO #<i> is not empty 1 FIFO #<i> is empty Mapping of these bit to an actual FIFO number is as follows: bit 0 => 0 bit 1=> 1 bit 2 => 2 bit 3 =>1c bit 4 => 2c bit 5 => 5b bit 6 => 5f bit 7 => 6b bit 8 => 6f bit 9 => 9 bit 10 => 10 (MCU access) bit 11 => 11 (MCU access)
11-0 DMFC_FIFO_FULL_<i>	This bit indicates on a DMFC FIFO#<i> full condition. 0 FIFO #<i> is not full 1 FIFO #<i> is full Mapping of these bit to an actual FIFO number is as follows: bit 0 => 0 bit 1=> 1 bit 2 => 2 bit 3 =>1c bit 4 => 2c bit 5 => 5b bit 6 => 5f bit 7 => 6b bit 8 => 6f bit 9 => 9 bit 10 => 10 (MCU access) bit 11 => 11 (MCU access)

42.2.3.14 VDI Registers

42.2.3.14.1 VDI Field Size Register (VDI_FSIZE)

The register used to control size of VDI input fields.

Address 0xBASE+0xE068000 (VDI_FSIZE) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	VDI_FHEIGHT									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	VDI_FWIDTH									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-405. VDI Field Size Register (VDI_FSIZE)

Table 42-408. VDI_FSIZE Register Description

Field	Description
31–26	Reserved
25–16 VDI_FHEIGHT	Frame height The value to be written to this register is the frame's height minus 1. The frame height should not be smaller than 16. When VDI_CMB_EN bit is clear: • The frame height should not be greater than 1080. • The frame's height must be even (which means that both fields have the same height) • The frame's height in 4:2:0 format, must be multiple of 4 (which means that both chroma fields have the same height) When VDI_CMB_EN bit is set: • The frame height should not be greater than 1200.
15–10	Reserved
9–0 VDI_FWIDTH	Frame width. The value to be written to this register is the frame's width minus 1. The Frame width should not be smaller than 16. The width must be even. When VDI_CMB_EN bit is clear • The Frame width should not be greater than 720. When VDI_CMB_EN bit is set: • The Frame width should not be greater than .

42.2.3.14.2 VDI Control Register (VDI_C)

The register used to control modes of operations of VDI module.

Address 0xBASE+0xE068004 (VDI_C) Access: User read-write

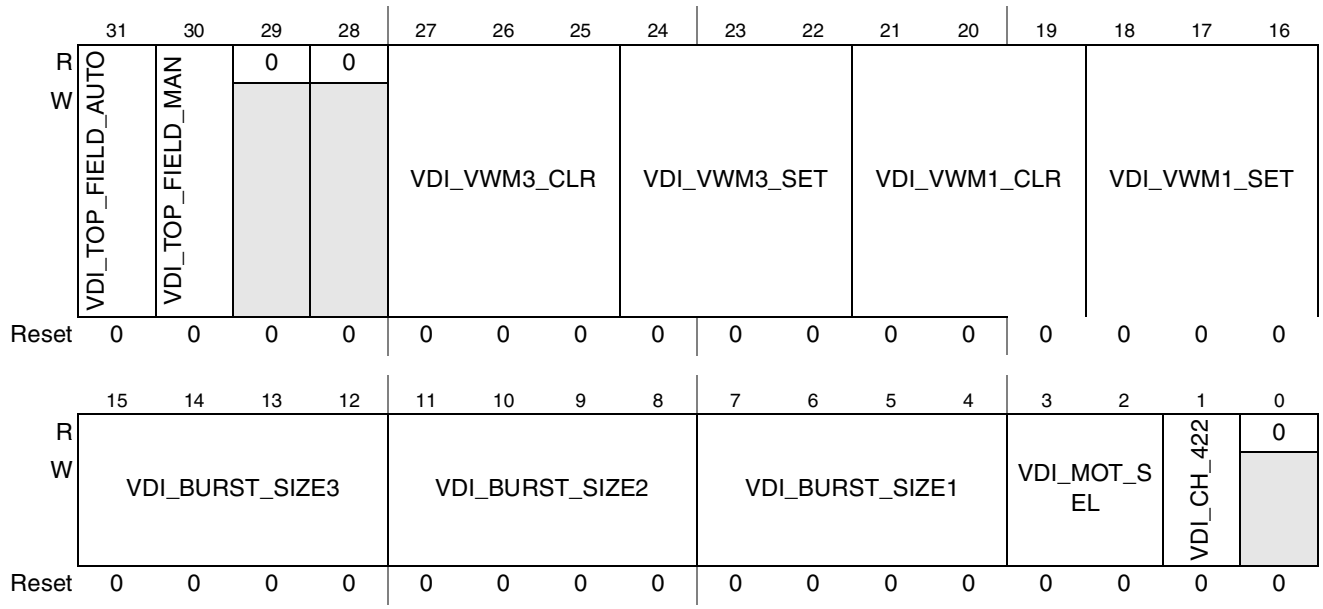


Figure 42-406. VDI Control Register (VDI_C)

Table 42-409. VDI_C Register Description

Field	Description
31	VDI top filed (automatic) This defines what would be the top field to be processed when the data is coming from the CSI 0 - top field is field 0 1- top field is field 1
30	VDI top filed (manual) This defines what would be the next top field to be processed when the data is coming from the memory 0 - top field is field 0 1- top field is field 1
29–28	Reserved
27-25 VDI_VWM3_CLR	VDI WaterMark “clear” level for channel 3. 0 clear watermark level when FIFO3 is full on 1/8 of their size. 1 clear watermark level when FIFO3 is full on 2/8 of their size. ... ... 7 clear watermark level when FIFO3 is full.
24-22 VDI_VWM3_SET	VDI WaterMark “set” level for channel 3. 0 set watermark level when FIFO3 is full on 1/8 of their size. 1 set watermark level when FIFO3 is full on 2/8 of their size. ... ... 7 set watermark level when FIFO3 is full.

Table 42-409. VDI_C Register Description (continued)

Field	Description
21-19 VDI_VWM1_CLR	VDI WaterMark “clear” level for channel 1 or channel 4 (channels 1 and 4 are not working simultaneously). 0 clear watermark level when FIFO1 is full on 1/8 of their size. 1 clear watermark level when FIFO1 is full on 2/8 of their size. ... ... 7 clear watermark level when FIFO1 is full.
18-16 VDI_VWM1_SET	VDI WaterMark “set” level for channel 1 or channel 2 (channels 1 and 4 are not working simultaneously). 0 set watermark level when FIFO1 is full on 1/8 of their size. 1 set watermark level when FIFO1 is full on 2/8 of their size. ... ... 7 set watermark level when FIFO1 is full.
15-12 VDI_BURST_SIZE3	Burst Size for channel 3. 0 Burst size is 1 access. 1 Burst size is 2 accesses. ... ... 15 Burst size is 16 accesses. The VDI’s burst size is restrict by the IDMAC’s restriction - This value must match the IDMAC settings and follow the IDMAC’s restrictions
11-8 VDI_BURST_SIZE2	Burst Size for channel 2. 0 Burst size is 1 access. 1 Burst size is 2 accesses. ... ... 15 Burst size is 16 accesses. The VDI’s burst size is restrict by the IDMAC’s restriction - This value must match the IDMAC settings and follow the IDMAC’s restrictions
7-4 VDI_BURST_SIZE1	Burst Size for channels 1 or 4 (channels 1 and 4 are not working simultaneously). 0 Burst size is 1 access. 1 Burst size is 2 accesses. ... ... 15 Burst size is 16 accesses. The VDI’s burst size is restrict by the IDMAC’s restriction - This value must match the IDMAC settings and follow the IDMAC’s restrictions
3-2 VDI_MOT_SEL	Motion select. 0 Motion determined by ROM “1” (shared toward medium/high motion). 1 Motion determined by ROM “2” (This option will not work well for high motion). 2 Full motion, only vertical filter is used 3 Forbidden.
1 VDI_CH_422	Chroma format at input and output of VDI. 0 Chroma format is 420. 1 Chroma format is 422.
0	Reserved

42.3 Functional Description

42.3.1 IPU Detailed Block Diagram

Figure 42-407 is the IPUv3EX top level block diagram.

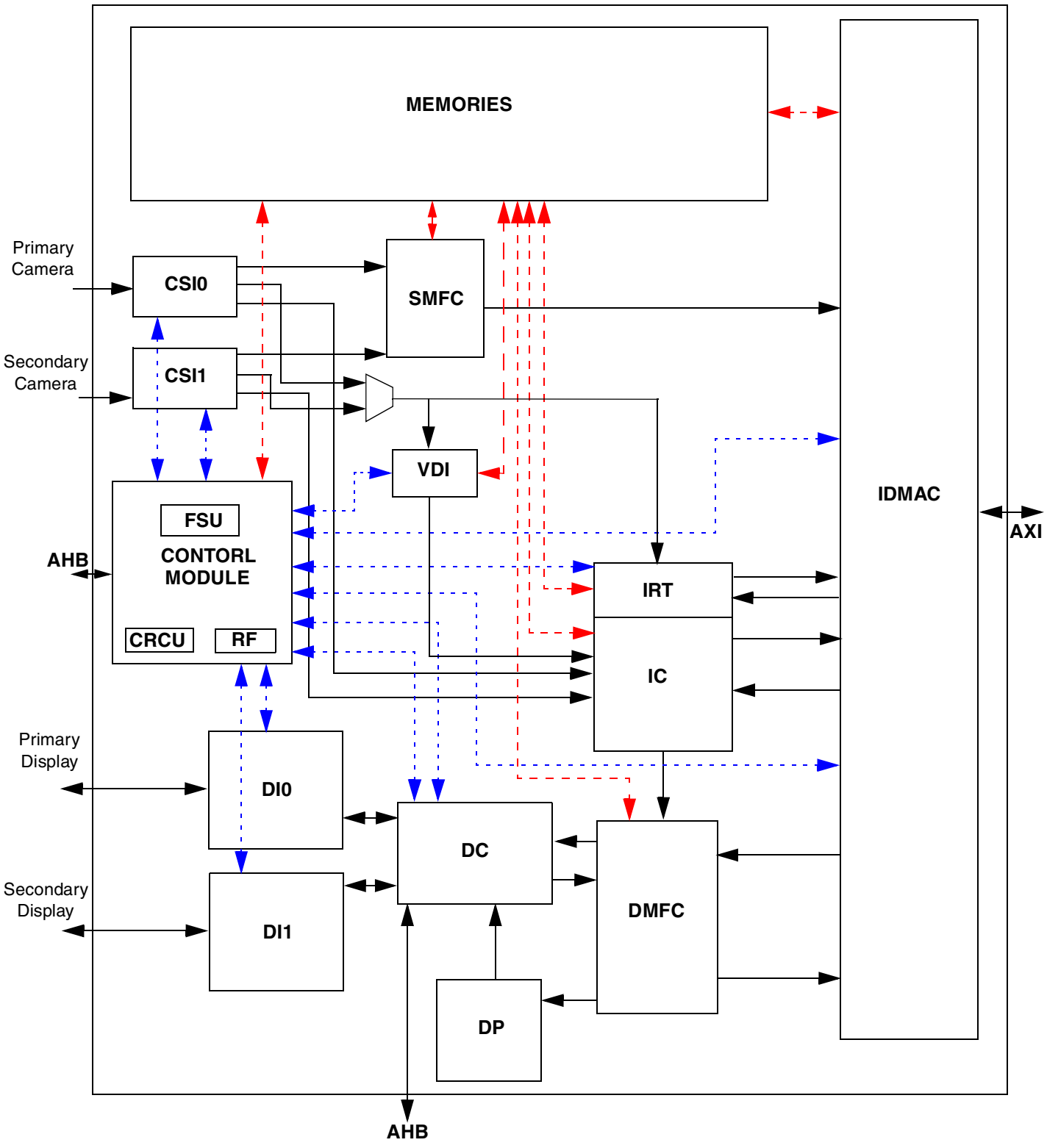


Figure 42-407. IPUv3EX Detailed Block Diagram

42.3.2 Image DMA Controller (IDMAC)

The following diagram is the IDMAC's block diagram.

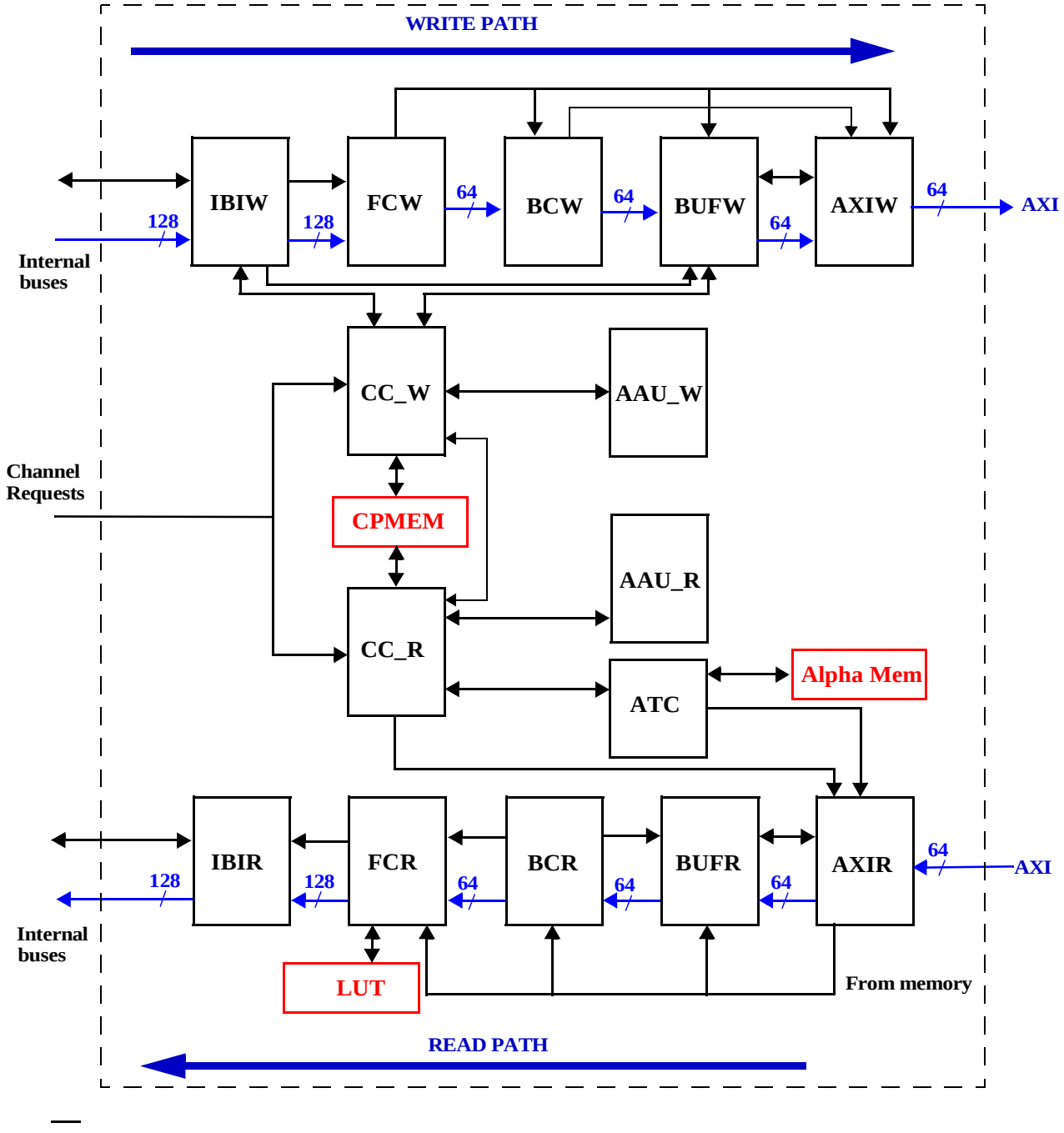


Figure 42-408. IDMAC Block Diagram

The following table describes the IDMAC's sub module glossary

Table 42-410. IDMAC's sub modules glossary

Sub Module	Description
IBIW	Internal Bus Interface Write
IBIR	Internal Bus Interface Read
FCW	Format Converter Write
FCR	Format Converter Read
BCW	Buffer Controller Write
BCR	Buffer Controller Read
BUFW	Buffer Write
BUFR	Buffer Read
AXIW	AXI Write
AXIR	AXI Read
CC_W	Channel Control Write
CC_R	Channel Control Read
AAU_W	Address Arithmetic Unit Write
AAU_R	Address Arithmetic Unit Read
ATC	Alpha Transparency Controller
LUT	Look up table
CPMEM	Channel Parameter Memory

42.3.2.1 IDMAC's channels

The table below summarizes the IDMAC's channels. Enabling a channel is done via the channel's corresponding IDMAC_CH_EN bit.

.

Table 42-411. IDMAC DMA channels list

Channel #	Source	Destination	Alternate Destination	Purpose	Data type
0	CSI (via SMFC)	Fmem		VF2 - Bayer; BPP>8; JPEG;	Generic or Pixel
1	CSI (via SMFC)	Fmem		VF2 - Bayer; BPP>8; JPEG;	Generic or Pixel
2	CSI (via SMFC)	Fmem		VF2 - Bayer; BPP>8; JPEG;	Generic or Pixel
3	CSI (via SMFC)	Fmem		VF2 - Bayer; BPP>8; JPEG;	Generic or Pixel

Table 42-411. IDMAC DMA channels list

Channel #	Source	Destination	Alternate Destination	Purpose	Data type
5	VDI	Bmem	IC	VF1/VF2	Pixel
8	Fmem	VDI		Previous field	Pixel
9	Fmem	VDI		Current field	Pixel
10	Fmem	VDI		Next field	Pixel
11	Bmem	IC		video plane for post processing task	Pixel
12	Bmem	IC		video plane for PrP tasks (view finder or encoding)	Pixel
13	VDI	Fmem		Recent field from CSI	Pixel
14	Fmem	IC		graphics plane for PrP task (view finder or encoding)	Pixel
15	Fmem	IC		graphics plane for post processing task	Pixel
16	Reserved				
17	Fmem	IC		Transparency (alpha for channel 14)	Generic
18	Fmem	IC		Transparency (alpha for channel 15)	Generic
19					
20	IC	Bmem		Preprocessing data from IC (encoding task) to memory	Pixel
21	IC	Bmem	DMFC	Preprocessing data from IC (viewfinder task) to memory; This channel can be configured to send the data directly to the DMFC. This is done by programming the IC_DMFC_SEL bit.	Pixel
22	IC	Bmem		Postprocessing data from IC to memory	Pixel
23	Fmem	DP		DP primary flow - main plane	Pixel
24	Fmem	DP		DP secondary flow - main plane	Pixel
25					
26					
27	Fmem	DP		DP primary flow - auxiliary plane	Pixel
28	Fmem	DC		DC channel for both sync and async flows	Pixel
29	Fmem	DP		DP secondary flow - auxiliary plane	Pixel
30	Reserved				
31	Fmem	DP		Transparency (alpha for channel 27)	Generic
32	Reserved				
33	Fmem	DP		Transparency (alpha for channel 29)	Generic
34	Reserved				
35	Reserved				
36	Reserved				

Table 42-411. IDMAC DMA channels list

Channel #	Source	Destination	Alternate Destination	Purpose	Data type
37	Reserved				
38	Reserved				
39	Reserved				
40	DC	Fmem		DC read channel	Generic
41	Fmem	DC		DC async flow	Generic
42	Fmem	DC		DC command stream	Generic
43	Fmem	DC		DC command stream	Generic
44	Fmem	DC		DC output mask	Generic
45	Bmem	IRT		Rotation for post Encoding task	Pixel
46	Bmem	IRT		Rotation for viewfinder task	Pixel
47	Bmem	IRT		Rotation for post processing task	Pixel
48	IRT	Bmem		Rotation for Encoding task	Pixel
49	IRT	Bmem		Rotation for viewfinder task	Pixel
50	IRT	Bmem		Rotation for post processing task	Pixel
51	Fmem	DP		Transparency (alpha for channel 23)	Generic
52	Fmem	DP		Transparency (alpha for channel 24)	Generic
53-63	Reserved				

42.3.2.2 IBIW & IBIR - Internal Bus Interface for Write and Read

The Internal Bus Interface handles the internal IPUv3EX protocol communicating between the IDMAC and the IPUv3EX's sub modules. The IBIR handles channels that perform read from external memory. The IBIW handles channels that perform write accesses to external memory.

42.3.2.3 FCW & FCR - Format Converter Write and Read

The Format Converter performs packing ("write direction") / unpacking ("read" direction) of pixels with programmable position and width of color components, decoding 4- or 8-bits coded pixels according to a loaded look-up table, panning of an image read from the system memory according to a panning offset (start pixel address). The Format Converter supports formats with a pixel width of 4, 8, 12, 16, 18, 24 or 32 bits. The format converter unit handles two pixels simultaneously.

The IPUv3EX sub modules can handle only the formats, presented below. Each component is 8 bit:

Figure 42-409. IPUv3EX **internal pixel formats**

The pixel can be stored in the memory in the formats presented below. (R/G/B means that this could component can be R or G or B; A is the location of the alpha component).

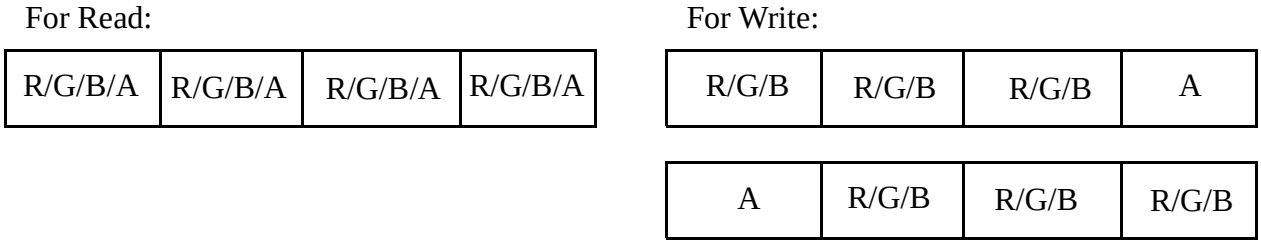
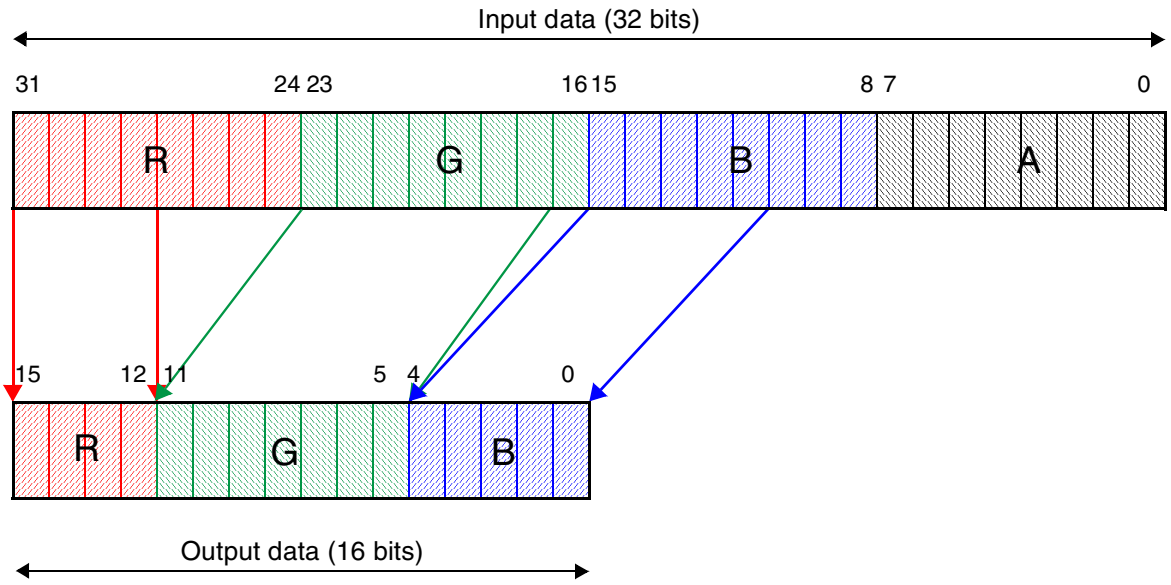


Figure 42-410. IPUv3EX **external pixel formats**

Formatting parameters are written in the Channel Parameter Memory (See: 42.3.2.10 CPMEM - Channel Parameter Memory). The following parameters are used:

- Offset OFS0 between MSB position of the color component 0 and MSB position of packed pixel. The color component 0 occupies the most significant bits of the unpacked pixel (mostly this is the R component). The OFS0 range is from 0 to 31.
- Color component 0 width WID0 minus 1.
- Offset OFS1 between MSB position of the color component 1 and MSB position of packed pixel. The color component 1 occupies the middle left bits of the unpacked pixel (mostly this is the G component). The OFS1 range is from 0 to 31.
- Color component 1 width WID1 minus 1.
- Offset OFS2 between MSB position of the color component 2 and MSB position of packed pixel. The color component 2 occupies the middle right bits of the unpacked pixel (mostly this is the B component). The OFS2 range is from 0 to 31.
- Color component 2 width WID2 minus 1.
- Offset OFS3 between MSB position of the color component 3 and MSB position of packed pixel. The color component 3 occupies the least significant bits of the unpacked pixel (mostly this is the A component). The OFS3 range is from 0 to 31. For write specified DMA channels, the OFS3 value is set to 24 or 0 bits.
- In cases of read with separate alpha (alpha is located in a separate buffer in the system’s memory than the pixel data), the alpha component size is defined according to WID3.

Figure 42-411 and Figure 42-412 show examples of data packing and unpacking.



Byte 0 (red) **OFS0 = 0** **WID0 = 3**
Byte 1 (green) **OFS1 = 4** **WID1 = 6**
Byte 2 (blue) **OFS2 = 11** **WID2 = 4**

Figure 42-411. Data Packing Example

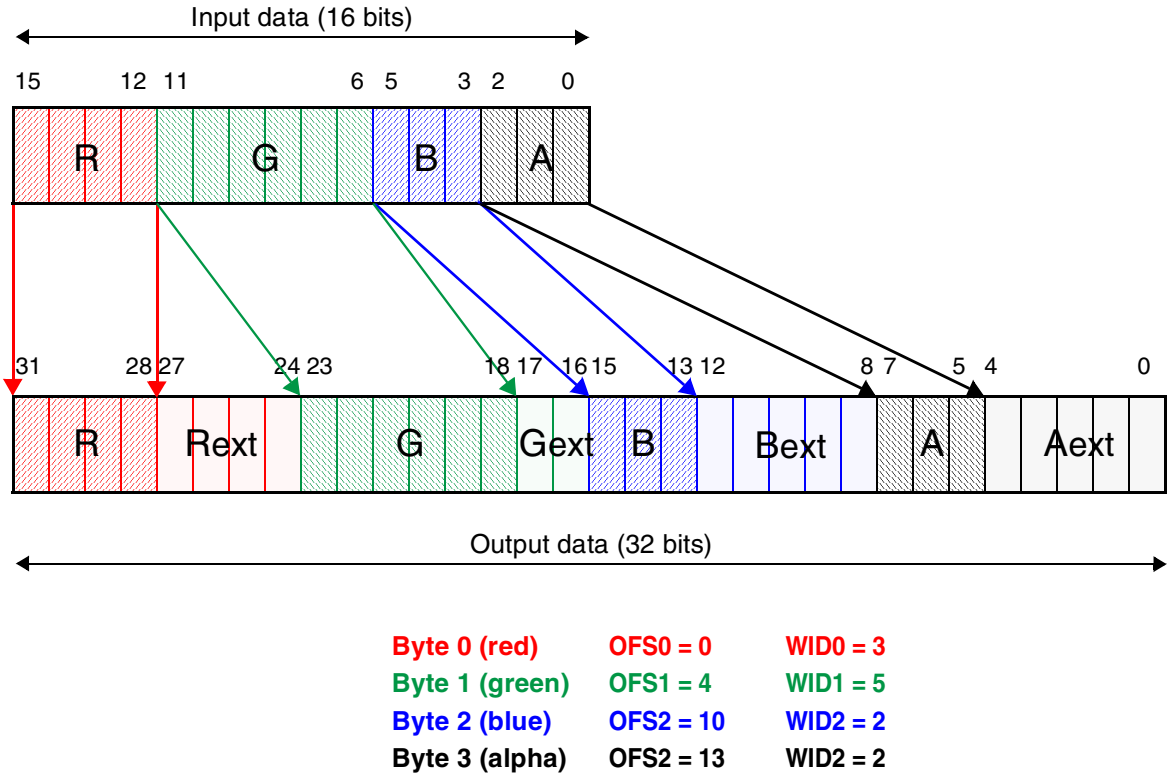


Figure 42-412. Data Unpacking Example

If read data has the coded format, it is decoded via the look-up table. The Look-Up Table Memory (See: 42.3.2.9 LUT- Look Up Table) must be loaded at the IDMAC initialization step. The LUT output format must match the IPUv3EX internal format RGBA 8888 where R is placed in MSB and A is placed in LSB. The A field is used only for graphics data.

42.3.2.4 Buffering units

The buffering units (BUFW, BUFR) are used to store the data consist of different coded color components

- On write transactions (BUFW), before writing to memory & after the format has been coded.
- On read transactions (BUFR), after reading from memory & before the format has been decoded.

Each buffering unit includes 4 X 64 bytes buffers. Each buffer, for each direction, can handle any kind of color component (interleaved - Y / partial interleaved - Y, UV / non interleaved - Y, U, V).

The Write buffers are controlled by the buffer controller for write (BCW) and AXIW units.

The Read buffers are controlled by the buffer controller for read (BCR) and AXIR units.

42.3.2.4.1 Handling real time channels

The memory controller connected to the AXI bus of the IPUv3M can use the AXI ID associated with each burst in order to distinguish between real time and non real time channels. In order to do that the user has to set the channel's ID according to the settings in the memory controller and set the priority of the channel according to its nature. The buffer controller (BCW/BCR) holds all the pending requests that won the arbitration. However, as the memory controller can distinguish between the real time channels and non real time channels within the IPU, there could be a situation where the real time requests are blocked as the IPU's queue is filled with non real time requests. In order to avoid that the user can limit the number of non-real time requests in the queue.

The queue for read requests can handle up to 8 requests. The queue for write requests can handle up to 6 requests. The user can limit the number of non real time requests by setting the USED_BUFS_MAX_W for write requests and USED_BUFS_MAX_R for read requests. The feature that limits the number of requests is enabled by setting the USED_BUFS_EN_R bit for the read requests and USED_BUFS_EN_W for the write requests.

42.3.2.5 AXIW - AXI Write and AXIR - AXI Read

The AXI Master Interfaces are responsible for data transfer from/to the system memory. The Interface supports only 64-bits burst accesses of 1-8 words, with nonalignment of a byte resolution.

2 separate & independent AXI masters are used for "read" (AXIR) & "write" (AXIW), each can be programmed (via CPMEM) with 4 different ID's to support out-of-order accesses within bursts.

42.3.2.6 CC_W & CC_R - Channel Control Write and Read

The Channel Control units is the main control unit of the IDMAC.

- It arbitrates the channels according to the priority.
- Control the address arithmetic unit.
- Function as a memory interface to the CPMEM. It reads the parameters from the CPMEM, prepare the controls accordingly and write back updated parameters to the CPMEM.
- The read unit provides the parameters to the IBIR unit
- The write unit provides the parameters to the AXIW unit

The CC calculates all the parameters related to the access except the address and BS (burst size) which are calculated at the AAU.

The priority is set according to:

- The channel's corresponding bit in the IDMAC_CH_PRI_1 & IDMAC_CH_PRI_2 registers.
- The watermark signal generated from the sub module. The watermark signal is ignored unless the channel's corresponding IDMAC_WM_EN bit is set.
- Special priority for alpha channels

A priority value is calculated for each of the enabled channels according to the above conditions. Then the CC units selects between the channels with the same priority value in a round-robin fashion.

Table 42-412. Calculated priority value

alpha channel	Channel's priority bit	watermark signal	Priority Value
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	1
1	0	1	2
1	1	0	3
1	1	1	4

42.3.2.7 AAU_W & AAU_R- Address Arithmetic Unit for Write and Read

The AAU_R & AAU_W units calculate the address in the system memory to be accessed by the IPUv3EX. These units also calculate the burst size (BS). The address calculation is done according to parameters stored in the CPMEM.

The following main addressing parameters are used:

- XB—Horizontal pixel position in frame
- YB—Vertical pixel position in frame
- SL—Stride line minus 1 (gap in bytes between two pixels in the same column in two consecutive rows).
- SX—Horizontal pixel scrolling offset
- SY—Vertical pixel scrolling offset
- EBA—Frame buffer base address in bytes (there are two such parameters to support double buffering)
- BPP—Bits per pixel
- FW—Frame width minus 1
- FH—Frame height minus 1

Relations between the addressing parameters and image frame are shown in Figure 42-413.

The system memory address in bytes is calculated as:

$$ADDR = EBA + (XB + SX) \cdot BPP + (YB + SY) \cdot (SL + 1)$$

- with 0 < XB =< FW and 0 < YB =< FH.
- For non-interleaved formats the 4 LSB bits of SX are defined according to the IOX parameter.

When double buffering is used, the EBA0 is the base address of the buffer 0 and the EBA1 is the base address of the buffer 1. The IPU_CHA_CUR_BUF Register is a status register. It contains 1-bit pointers to the current working buffers for all IPUv3EX DMA channels. The IPUv3EX automatically toggles a pointer after completion of the current buffer processing. If the MCU is a data source for specific double-buffered channel, it should check this status bit in order to know what is the IPUv3EX current buffer. The MCU is allowed to write to the buffer only when a working DMA channel does not use it. After the MCU has been fill the buffer, it has to set the corresponding bit in the IPU_CHA_BUF0_RDY and IPU_CHA_BUF1_RDY Registers. If needed, the MCU can only clear the pointer by writing 1 but not set it.

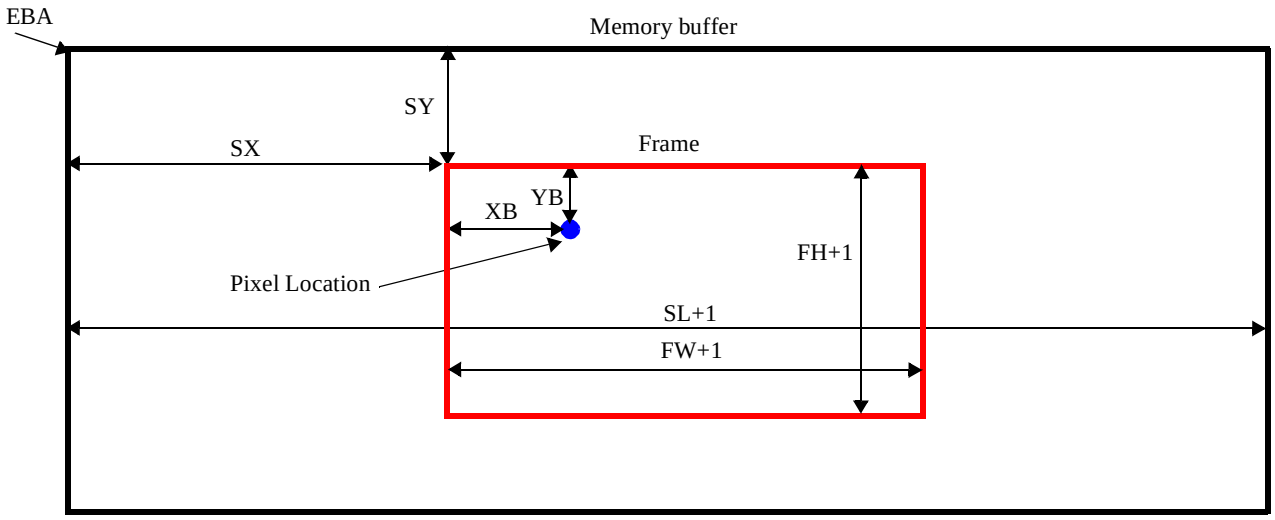


Figure 42-413. Addressing Parameters and Image Frame

The XB and YB coordinates are calculated according to addressing mode. There are two addressing modes:

- 2D mode
- Block mode

In 2D mode the pixel data is transferred to the memory row-by-row. There are two ways to use 2D mode: start from YB = 0 and finish at YB =< FH (YB is incremented) or start from YB = FH and finish at YB =< 0 (YB is decremented). The second option provides vertical flip of the image.

In block mode the frame is divided into blocks. This is needed for rotation or post-filtering, where the order used for data transfers is block-by-block. The order of the block transfer is according to the VF,HF and ROT bits in the IDMAC Channel Parameter Memory. The order within the block is row-by-row where the block size is limited by the block width (BW) and block height (BH) parameters. The BW and BH parameters are set by IC rotation section and cannot be configured through the Channel Parameter Memory.

The Channel Control is responsible for the address calculation flow. It takes channel parameters from the Channel Parameter Memory, updates them and controls the Address Arithmetic Unit.

42.3.2.7.1 Scrolling support

Automatic display of a changing image (animation) or moving image (scrolling) is implemented by reading frames (from a background buffer) with incremental offset. Enabling the scrolling feature is done by setting the channel's corresponding SCE bit. The scrolling step is controlled by channel's corresponding SDX and SDY parameters. The scrolling direction is defined by the channel's corresponding SDRX and SDRY parameters. The maximum number of scrolled frames to be read is defined by the channel's corresponding SM parameter.

When the last programmed frame is reached (IDMAC's internal counter reached SM), IDMAC can perform one of the following (controlled by the SCC bit):

- Return to the first frame, without any SW intervention. The return point is defined by SX0 and SY0 parameters.
- Interrupt the MCU, to generate the next content.

42.3.2.8 ATC - Alpha Transparency Controller

The Alpha transparency controller (ATC) handles the alpha buffers on the external memory for cases where the pixel data and the alpha data are located on separate buffers (separate alpha mode). In that case the IDMAC reads the alpha data and the pixel data, merge them together and provides a pixel that includes the alpha information to the relevant sub module. The ATC's main functions are:

- Generates requests for alpha channels, following a request to pixel data from the module.
- Maintain an internal alpha memory buffer for each channel.
- When there isn't enough alphas in the memory the ATC blocks the corresponding pixel channels.
- When there is a request of pixel channel the ATC load and accumulate its alphas in a register.
- ATC memory controller can manage 8 channels of alphas.
- ATC supports synchronous new frame before end of frame errors.

In order to configure the channel to use separate alpha the channel's corresponding IDMAC_SEP_AL bit should be set in addition to the ALU bit in the CPMEM of the corresponding channel.

Figure 42-414 is the ATC's block diagram.

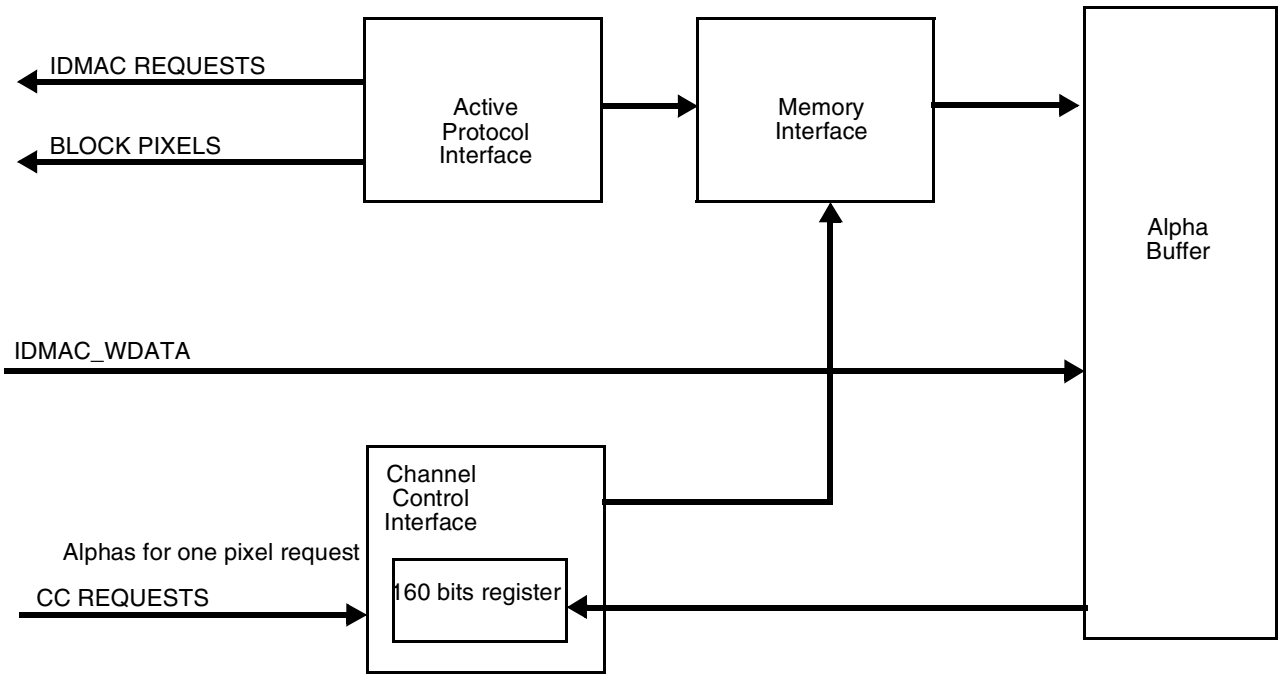


Figure 42-414. ATC block diagram

The ATC alpha buffer memory can hold up to 8 buffers of alphas. A pointer to a buffer in the ATC memory is defined according to the ALBM parameter in the CPMEM. Table 42-413 describes the relations between a data channel, an alpha channel and the pointer in the alpha buffer mmeory.

Table 42-413. Alpha channels mapping

data channel number	associated alpha channel number	Alpha buffer memory (ALBM)
14	17	0
15	18	1
27	31	2
29	33	3
23	51	4
24	52	5

42.3.2.8.1 Conditional read

The alpha data can be used to reduce reads from the memory of pixels that are going to be transparent (alpha = 0). The conditional read feature is enabled by the CRE bit in the CPMEM. If all the corresponding alpha values for a single burst of pixels are all equal to zero, the IDMAC will block the access to the external memory and provide a data of all zeros to the corresponding channel. This way some of the accesses to the memory can be prevented thus reducing the load on the memory.

42.3.2.9 LUT- Look Up Table

When working in coded pixel format, the data read from the memory is the decoded value of pixel according to address given. In case of 8 bit code, the data read from the memory is the decoded value of the pixel according to the address given.

In case of 4 bit code configuration, The address of the 4 bit decoded values is set according to DEC_SEL parameter in the CPMEM

00 = addresses 0 to 15

01 = addresses 64 to 79

10 = addresses 128 to 143

11 = addresses 192 to 207

Table 42-414. Look-Up Table Memory Structure

Address	Word	DEC_SEL	Description
0	Word0	4 BPP = 00 8 BPP = don't care	Decoded Pixels [15:0] for both 8 and 4 bit coded configuration
...	...		
15	Word15		
16	Word16	don't care	Decoded Pixels [63:16] for 8 bit coded configuration only
...	...		
63	Word63		
64	Word64	4 BPP = 01 8 BPP = don't care	Decoded Pixels [79:64] for both 8 and 4 bit coded configuration
...	...		
79	Word79		
80	Word80	don't care	Decoded Pixels [127:80] for 8 bit coded configuration only
...	...		
127	Word127		

Table 42-414. Look-Up Table Memory Structure (continued)

Address	Word	DEC_SEL	Description
128	Word128	4 BPP = 10 8 BPP = don't care	Decoded Pixels [143:128] for both 8 and 4 bit coded configuration
...	...		
143	Word143		
144	Word144	don't care	Decoded Pixels [191:144] for 8 bit coded configuration only
...	...		
191	Word191		
192	Word192	4 BPP = 11 8 BPP = don't care	Decoded Pixels [207:192] for both 8 and 4 bit coded configuration
...	...		
207	Word207		
208	Word208	don't care	Decoded Pixels [255:208] for 8 bit coded configuration only
...	...		
255	Word255		

42.3.2.10 CPMEM - Channel Parameter Memory

The CPMEM holds the configuration parameters for each IDMAC channel. The CPMEM can holds the settings of 80 channels. Each channel's settings are defined by a two mega-words. Each mega-word is 160 bits wide. The following diagram illustrates the CPMEM's structure.

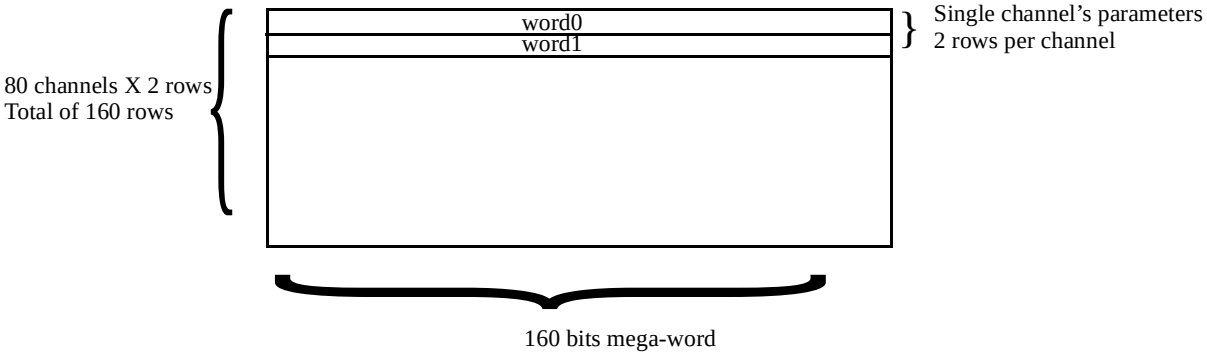


Figure 42-415. CPMEM structure

Each IDMAC channel can be configured to work in one of two different modes:

- Non Interleaved mode where the Y:U:V data is organized in 3 separate buffers in the system's memory
- Interleaved mode where the Y:U:V data is organized in a single buffer in the system's memory

The parameters and the way they are organized are different for each mode (interleaved or non-interleaved).

The following tables describe the IDMAC parameters and their organization in each mode.

42.3.2.10.1 CPMEM's words' structure for non interleaved mode

The table below describes the CPMEM's words' structure for non interleaved mode. Each CPMEM word consist of 160 bits. The bits that are not listed in the table below are reserved bits

.

Name	Mnemonic	Size	Location	Description
Word 0				
XV Virtual Coordinate	XV	10 bits	W0[9:0]	Variable coordinates for determining next block address. {X1,Y1} and {X2,Y2} coordinates will be determined according to {XV,YV} upon restart of channel. These coordinates are used for Y:U:V (Y pointer) and RGB formats.
YV Virtual Coordinate	YV	9 bits	W0[18:10]	
XB inner Block Coordinate	XB	13 bits	W0[31:19]	Variable coordinates for determining address within the block. These coordinates are used for Y:U:V (Y pointer) and RGB formats. Need 24 bits for 2D transfer support.
YB inner Block Coordinate	YB	12 bits	W0[43:32]	
New Sub Block	NSB_B	1 bit	W0[44]	This bit determines if the next value for {XB,YB} should be taken from {XB,YB} saved in channel parameter memory or from new {x1,y1}/{x2,y2}.
Current Field	CF	1 bit	W0[45]	CF = 0 Current field is even CF = 1 Current field is odd
Mem U Buffer Offset	UBO	22 bits	W0[67:46]	Double buffer destination address offset for Y:U:V (U pointer) formats.The actual physical address value is divided by 8 (i.e. this parameter includes bits [24:3] of the actual address)
Mem V Buffer Offset	VBO	22 bits	W0[89:68]	Double buffer destination address offset for Y:U:V (V pointer) format. The actual physical address value is divided by 8 (i.e. this parameter includes bits [24:3] of the actual address)
Initial Offset X	IOX	4 bits	W0[93:90]	The IOX parameter, is the offset in pixels for a frame that starts at a non aligned address. for 42x formats must be even.
Reduce Double Read or Writes	RDRW	1 bits	W0[94:94]	This bit is relevant for YUV4:2:0 formats. For read channels: U and V components are not read from odd rows. (read - supported only for the VDI) For write channels: U and V components are not written to odd rows. (write - supported for all write channels)
Scan Order	SO	1 bit	W0[113]	SO = 0 Scan order is progressive SO = 1 Scan order is interlaced

Band Mode	BNDM	3 bits	W0[116:114]	BNDM = 000 bands disable. BNDM = 001 bands enable. Band height = 4 lines. BNDM = 010 bands enable. Band height = 8 lines. BNDM = 011 bands enable. Band height = 16 lines. BNDM = 100 bands enable. Band height = 32 lines. BNDM = 101 bands enable. Band height = 64 lines. BNDM = 110 bands enable. Band height = 128 lines. BNDM = 111 bands enable. Band height = 256 When working in band mode, the channel's corresponding IDMAC_BNDM_EN bit has to be set.
Block Mode	BM	2 bits	W0[118:117]	BM = 00 block mode disable. BW = FW, BH = FH BM = 01 block mode enable. BW = 8, BH = 8 BM = 10 block mode enable. BW = 16, BH = 16 (this mode is reserved for future use) BM = 11 not used
Rotation	ROT	1 bit	W0[119]	ROT = 0 -> No rotation ROT = 1 -> 90 degree rotation clockwise
Horizontal Flip	HF	1 bit	W0[120]	HF = 0 -> No flip HF = 1 -> Horizontal flip enable
Vertical Flip	VF	1 bit	W0[121]	VF = 0 -> No flip VF = 1 -> Vertical flip enable
Threshold Enable	THE	1 bit	W0[122]	THE = 0 -> Threshold disable THE = 1 -> Threshold enable
Conditional Access Polarity	CAP	1 bit	W0[123]	CAP = 0 -> If conditional bit in CM register is low skip the access CAP = 1 -> If conditional bit in CM register is high skip the access. This mode is reserved for future use.
Conditional Access Enable	CAE	1 bit	W0[124]	CAE = 0 -> Conditional access disable CAE = 1 -> Conditional access enable This mode is reserved for future use.
Frame Width	FW	13 bits	W0[137:125]	Number of pixels in one row, of the channel frame. FW 000000000000 = 0001 pixels 000000000001 = 0002 pixels ..... 111111111111 = 8192 pixels

Frame Height	FH	12 bits	W0[149:138]	Number of pixels in one column, of the channel frame. FH 000000000000 = 0001 line 000000000001 = 0002 lines ..... 111111111111 = 4096 lines For progressive YUV 4:2:0 (non interleaved and partial interleaved formats) the FH value should be a multiple of 2. For interlaced YUV 4:2:0 (non interleaved and partial interleaved formats) the FH value should be a multiple of 4.
Word 1				
Ext Mem Buffer 0 Address	EBA0	29 bits	W1[28:0]	1st double buffer destination address for RGB and Y:U:V (Y pointer) formats. This parameter must not be changed if the corresponding channel is enabled. The actual physical address value is divided by 8 (i.e. this parameter includes bits [31:3] of the actual address)
Ext Mem Buffer 1 Address	EBA1	29 bits	W1[57:29]	2nd double buffer destination address for RGB and Y:U:V (Y pointer) formats. This parameter must not be changed if the corresponding channel is enabled. The actual physical address value is divided by 8 (i.e. this parameter includes bits [31:3] of the actual address)
Interlace Offset	ILO	20 bits	W1[77:58]	2nd double buffer destination address for RGB and Y:U:V (Y pointer) formats. An interlaced data stored in the memory can be read as a consecutive progressive only if FW*BPP is a multiplication of 8. The actual physical address value is divided by 8 (i.e. this parameter includes bits [22:3] of the actual address). For YUV420 formats, the ILO is relevant only to the Y component as the U and V components do not exist for the even lines. This value is signed

Number of Pixels in Whole Burst Access	NPB	7 bits	W1[84:78]	Number of pixels per burst access. The following are valid numbers of pixels in a memory burst access according to the BPP parameter: NPB 0000000 = 01 pixels in each burst 0000001 = 02 pixels in each burst ..... 1111111 = 128 pixels in each burst Range: 32BPP => 1 -> 16 pixels 24BPP => 1 -> 20 pixels 16BPP => 1 -> 32 pixels 12BPP => 1 -> 40 pixels 08BPP => 1 -> 64 pixels 04BPP => 1 -> 128 pixels
Pixel Format Select	PFS	4 bits	W1[88:85]	4'h0 = non-interleaved 4:4:4 4'h1 = non-interleaved 4:2:2 4'h2 = non-interleaved 4:2:0 4'h3 = partial interleaved 4:2:2 4'h4 = partial interleaved 4:2:0 4'h5 to 4'hF = NA
Alpha Used	ALU	1 bit	W1[89]	1 = the alpha associated with the data of this channel resides on another channel (separate buffer) 0 = the alpha associated with the data of this channel resides along with the pixel data (same buffer) The corresponding alpha channel must be enabled to assure correct behavior.
Alpha Channel Mapping	ALBM	2 bits	W1[92:90]	Alpha channel mapping – This parameter is a pointer to a buffer in the ATC memory. This parameter is relevant only to data channels that are associated with a separate alpha buffer (like graphic plane channels). The parameter should be programmed on the data channels' ALBM. Setting this parameter to any other channel has no meaning. See Table 42-413 for exact ALBM mapping.
AXI Id	ID	2 bits	W1[94:93]	AXI protocol id
Threshold	TH	7 bits	W1[101:95]	0000000 = 32 lines 0000001 = 64 lines ..... 1111111 = 4096 lines

Stride Line	SLY	14 bits	W1[115:102]	Address vertical scaling factor in bytes for memory access. Also number of maximum bytes in the “Y” component row according to memory limitations. SLY 00000000000000 = 00001 bytes 00000000000001 = 00002 bytes ..... 11111111111111 = 16384 bytes
Width3	WID3	3 bits	W1[127:125]	Fourth color component size of the input-unpacking/ output-packing pixel. WID3 000 = 1 bits 001 = 2 bits ..... 111 = 8 bits As this is a non-interleaved format, this field is relevant only to the alpha associated with this pixel channel.
Stride Line	SLUV	14 bits	W1[141:128]	Address vertical scaling factor in bytes for memory access. Also number of maximum bytes in the “U” or “V” component row according to memory limitations. SLUV 00000000000000 = 00001 bytes 00000000000001 = 00002 bytes ..... 11111111111111 = 16384 bytes
Conditional Read Enable	CRE	1 bit	W1[149:149]	This bit enables the conditional read feature.

Table 42-415. Channel Parameters Memory for non-interleaved

42.3.2.10.2 CPMEM’s words’ structure for interleaved mode

The table below describes the CPMEM’s words’ structure for interleaved mode. Each CPMEM word consist of 160 bits. The bits that are not listed in the table below are reserved bits

Name	Mnemonic	Size	Location	Description
Word 0				
XV Virtual Coordinate	XV	10 bits	W0[9:0]	Variable coordinates for determining next block address. {X1,Y1} and {X2,Y2} coordinates will be determined according to {XV,YV} upon restart of channel. These coordinates are used for Y:U:V (Y pointer) and RGB formats.
YV Virtual Coordinate	YV	9 bits	W0[18:10]	
YB inner Block Coordinate	XB	13 bits	W0[31:19]	Variable coordinates for determining address within the block. These coordinates are used for Y:U:V (Y pointer) and RGB formats. Need 24 bits for 2D transfer support.
XB inner Block Coordinate	YB	12 bits	W0[43:32]	

New Sub Block	NSB_B	1 bit	W0[44]	This bit determines if the next value for {XB,YB} should be taken from {XB,YB} saved in channel parameter memory or from new {x1,y1}/{x2,y2}.
Current Field	CF	1 bit	W0[45]	CF = 0 Current field is even CF = 1 Current field is odd
Scroll X counter	SX	12 bits	W0[57:46]	Holds the temporary count for the Scroll X in between frame For interleaved YUV4:2:2 formats the SX should be a multiple of 2.
Scroll Y counter	SY	11 bits	W0[68:58]	Holds the temporary count for the Scroll Y in between frame
Number of Scroll	NS	10 bits	W0[78:69]	This variable holds the total number of Scrolls
Scroll Delta X	SDX	7 bits	W0[85:79]	Frame start row offset, compared to last frame. SDX 0000000 = 00 pixels 0000001 = 01 pixels 0000010 = 02 pixels ..... 1111110 = 30 pixels 1111111 = 127 pixels For interleaved YUV4:2:2 formats the SDX should be a multiple of 2.
Scroll Max	SM	10 bits	W0[95:86]	Frame maximum row and column increment offset in frame. SM 0000000000 = 0001 0000000001 = 0002 ..... 1111111111 = 1024
Scrolling Configuration	SCC	1 bit	W0[96]	Determines if scrolling will continue from zero when NS counter has reached SM or stop at the current value for SX and SY for the next frames to come. SCC 0 => Scrolling will stop at NS = SM 1 => Scrolling will start from “0” at NS = SM
Scrolling Enable	SCE	1 bit	W0[97]	SCE = 0 Scrolling disable SCE = 1 Scrolling enable
Scroll Delta Y	SDY	7 bits	W0[104:98]	Frame start column offset, compared to last frame. SDY 0000000 = 00 pixels 0000001 = 01 pixels 0000010 = 02 pixels ..... 1111110 = 30 pixels 1111111 = 127 pixels

Scroll Horizontal Direction	SDRX	1 bit	W0[105]	Determines if the next frame will move right or left compared to the current frame. SDRX 0 => Next frame will be right of current 1 => Next frame will be left of current
Scroll Vertical Direction	SDRY	1 bit	W0[106]	Determines if the next frame will move down or up compared to the current frame. SDRY 0 => Next frame will be down of current 1 => Next frame will be up of current
Bits Per Pixel	BPP	3 bits	W0[109:107]	3'h0 = 32 Bits per pixel 3'h1 = 24 Bits per pixel 3'h2 = 18 Bits per pixel 3'h3 = 16 Bits per pixel 3'h4 = 12 Bits per pixel 3'h5 = 08 Bits per pixel 3'h6 = 04 Bits per pixel
Decode Address Select	DEC_SEL	2 bits	W0[111:110]	Upon 4BPP, selects between two look-up tables DEC_SEL 00 = addresses 0 to 15 01 = addresses 64 to 79 10 = addresses 128 to 143 11 = addresses 192 to 207
Access Dimension	DIM	1 bit	W0[112]	DIM = 0 Access Dimension is 2d DIM = 1 Access Dimension is 1d
Scan Order	SO	1 bit	W0[113]	SO = 0 Scan order is progressive SO = 1 Scan order is interlaced
Band Mode	BNDM	3 bits	W0[116:114]	BNDM = 000 bands disable. BNDM = 001 bands enable. Band height = 4 lines. BNDM = 010 bands enable. Band height = 8 lines. BNDM = 011 bands enable. Band height = 16 lines. BNDM = 100 bands enable. Band height = 32 lines. BNDM = 101 bands enable. Band height = 64 lines. BNDM = 110 bands enable. Band height = 128 lines. BNDM = 111 bands enable. Band height = 256 When working in band mode, the channel's corresponding IDMAC_BNDM_EN bit has to be set.

Block Mode	BM	2 bits	W0[118:117]	BM = 00 block mode disable. BW = FW, BH = FH BM = 01 block mode enable. BW = 8, BH = 8 BM = 10 block mode enable. BW = 16, BH = 16 (this mode is reserved for future use) BM = 11 not used
Rotation	ROT	1 bit	W0[119]	ROT = 0 -> No rotation ROT = 1 -> 90 degree rotation clockwise
Horizontal Flip	HF	1 bit	W0[120]	HF = 0 -> No flip HF = 1 -> Horizontal flip enable
Vertical Flip	VF	1 bit	W0[121]	VF = 0 -> No flip VF = 1 -> Vertical flip enable
Threshold Enable	THE	1 bit	W0[122]	THE = 0 -> Threshold disable THE = 1 -> Threshold flip enable
Conditional Access Polarity	CAP	1 bit	W0[123]	CAP = 0 -> If conditional bit in CM register is low skip the access CAP = 1 -> If conditional bit in CM register is high skip the access
Conditional Access Enable	CAE	1 bit	W0[124]	CAE = 0 -> Conditional access disable CAE = 1 -> Conditional access enable
Frame Width	FW	13 bits	W0[137:125]	Number of pixels in one row, of the channel frame. FW 000000000000 = 0001 pixels 000000000001 = 0002 pixels ..... 111111111111 = 8192 pixels For interleaved YUV4:2:2 formats the FW should be a multiple of 2.
Frame Height	FH	12 bits	W0[149:138]	Number of pixels in one column, of the channel frame. FH 000000000000 = 0001 line 000000000001 = 0002 lines ..... 111111111111 = 4096 lines
Word 1				
Ext Mem Buffer 0 Address	EBA0	29 bits	W1[28:0]	1st double buffer destination address for RGB and Y:U:V (Y pointer) formats. This parameter must not be changed if the corresponding channel is enabled.The actual physical address value is divided by 8 (i.e. this parameter includes bits [31:3] of the actual address)

Ext Mem Buffer 1 Address	EBA1	29 bits	W1[57:29]	2nd double buffer destination address for RGB and Y:U:V (Y pointer) formats. This parameter must not be changed if the corresponding channel is enabled.The actual physical address value is divided by 8 (i.e. this parameter includes bits [31:3] of the actual address)
Interlace Offset	ILO	20 bits	W1[77:58]	2nd double buffer destination address for RGB and Y:U:V (Y pointer) formats. An interlaced data stored in the memory can be read as a consecutive progressive only if FW*BPP is a multiplication of 8.The actual physical address value is divided by 8 (i.e. this parameter includes bits [22:3] of the actual address). This value is signed
Number of Pixels in Whole Burst Access	NPB	7 bits	W1[84:78]	Number of pixels per burst access. The following are valid numbers of pixels in a memory burst access according to the BPP parameter: NPB 0000000 = 01 pixels in each burst 0000001 = 02 pixels in each burst ..... 1111111 = 128 pixels in each burst Range: 32BPP => 1 -> 16 pixels 24BPP => 1 -> 20 pixels 16BPP => 1 -> 32 pixels 12BPP => 1 -> 40 pixels 08BPP => 1 -> 64 pixels 04BPP => 1 -> 128 pixels
Pixel Format Select	PFS	4 bits	W1[88:85]	4'h0 to 4'h4 = NA 4'h5 = Code (LUT) 4'h6 = Generic data 4'h7 = RGB (& also YUV interleaved 4:4:4) 4'h8 = interleaved 4:2:2 Y1U1Y2V1[1] 4'h9 = interleaved 4:2:2 Y2U1Y1V1[2] 4'hA = interleaved 4:2:2 U1Y1V1Y2[3] 4'hB = interleaved 4:2:2 U1Y2V1Y1[4] 4'hC to 4'hF = NA
Alpha Used	ALU	1 bit	W1[89]	1 = the alpha associated with the data of this channel resides on another channel (separate buffer) 0 = the alpha associated with the data of this channel resides along with the pixel data (same buffer) The corresponding alpha channel must be enabled to assure correct behavior.

Alpha Channel Mapping	ALBM	3 bits	W1[92:90]	Alpha channel mapping – This parameter is a pointer to a buffer in the ATC memory. This parameter is relevant only to data channels that are associated with a separate alpha buffer (like graphic plane channels). The parameter should be programmed on the data channels' ALBM. Setting this parameter to any other channel has no meaning. See Table 42-413 for exact ALBM mapping.
AXI Id	ID	2 bits	W1[94:93]	AXI protocol id; IPUv3EX is targeted to an AXI slave that can handle up to 2 requests with 2 different IDs + one request with a third ID. In case that IPUv3EX is going to be used on a system that can handle more than 2 requests with different IDs, the number of different IDs programmed in the CPMEM for different channels is limited for 2. This limitation is relevant for read channels only. For write channels there's no such limitation
Threshold	TH	7 bits	W1[101:95]	0000000 = 32 lines 0000001 = 64 lines ..... 1111111 = 4096 lines
Stride Line	SL	14 bits	W1[115:102]	Address vertical scaling factor in bytes for memory access. Also number of maximum bytes in row according to memory limitations. SL 00000000000000 = 00001 bytes 00000000000001 = 00002 bytes ..... 11111111111111 = 16384 bytes
Width0	WID0	3 bits	W1[118:116]	First color component size of the input-unpacking/ output-packing pixel. WID0 000 = 1 bits 001 = 2 bits ..... 111 = 8 bits This field is relevant only for interleaved RGB format (PFS = 4'h7)
Width1	WID1	3 bits	W1[121:119]	Second color component size of the input-unpacking/ output-packing pixel. WID1 000 = 1 bits 001 = 2 bits ..... 111 = 8 bits This field is relevant only for interleaved RGB format (PFS = 4'h7)

Width2	WID2	3 bits	W1[124:122]	Third color component size of the input-unpacking/ output-packing pixel. WID2 000 = 1 bits 001 = 2 bits ..... 111 = 8 bits This field is relevant only for interleaved RGB format (PFS = 4'h7)
Width3	WID3	3 bits	W1[127:125]	Fourth color component size of the input-unpacking/ output-packing pixel. WID3 000 = 1 bits 001 = 2 bits ..... 111 = 8 bits This field is relevant only for interleaved RGB format (PFS = 4'h7)
Offset0	OFS0	5 bits	W1[132:128]	Number of bits between MSB of pixel and MSB of color component, on input. 1 states that the color component will be the first color component aligned to MSB of output pixel. OFS0 00000 = No offset 00001 = u => 1 bit sleft, p => 1 bit sright ..... 11111 = u => 31 bit sleft, p => 31 bit sright * u = unpacking * p = packing * sleft = shift left * sright = shift right This field is relevant only for interleaved RGB format (PFS = 4'h7)
Offset1	OFS1	5 bits	W1[137:133]	Number of bits between MSB of pixel and MSB of color component on input. 2 states that the color component will be the second color component aligned to MSB output pixel. OFS1 00000 = No offset 00001 = u => 1 bit sleft, p => 1 bit sright ..... 11111 = u => 31 bit sleft, p => 31 bit sright * u = unpacking * p = packing * sleft = shift left * sright = shift right This field is relevant only for interleaved RGB format (PFS = 4'h7)

Offset2	OFS2	5 bits	W1[142:138]	Number of bits between MSB of pixel and MSB of color component on input. 3 states that the color component will be the third color component aligned to MSB output pixel. OFS2 00000 = No offset 00001 = u => 1 bit sleft, p => 1 bit sright ..... 11111 = u => 31 bit sleft, p => 31 bit sright * u = unpacking * p = packing * sleft = shift left * sright = shift right This field is relevant only for interleaved RGB format (PFS = 4'h7)
Offset3	OFS3	5 bits	W1[147:143]	Number of bits between MSB of pixel and MSB of color component on input. 4 states that the color component will be the fourth color component aligned to MSB output pixel. OFS3 00000 = No offset 00001 = u => 1 bit sleft, p => 1 bit sright ..... 11111 = u => 31 bit sleft, p => 31 bit sright * u = unpacking * p = packing * sleft = shift left * sright = shift right This field is relevant only for interleaved RGB format (PFS = 4'h7)
Select SX SY Set	SXYS	1 bit	W1[148:148]	This bit selects between the settings on: SC_CORD and SC_CORD1
Conditional Read Enable	CRE	1 bit	W1[149:149]	This bit enables the conditional read feature.
Decode Address Select bit[2]	DEC_SEL2	1 bit	W1[150:150]	This field is reserved

[1] Y1U1Y2V1 means byte0 = bits [7:0] =Y1; byte1 = bits [15:8] =U1; byte2 = bits [23:16] =Y2; byte3 = bits [31:24] = V1

[2] Y2U1Y1V1 means byte0 =Y2; byte1 =U1; byte2 =Y1; byte3 = V1

[3] U1Y1V1Y2 means byte0 =U1; byte1 =Y1; byte2 =V1; byte3 = Y2

[4] U1Y2V1Y1 means byte0 =U1; byte1 =Y2; byte2 =V1;byte3 = Y1

Table 42-416. Channel Parameters Memory for interleaved

42.3.2.10.3 Accessing the CPMEM for programming

Each IDMAC's channel's parameters are located on 2 CPMEM entries. Each Entry is 160 bit. The CPMEM is memory mapped and accessible via the AHB bus. The AHB bus is accesses are 32bit wide. A CPMEM entry is composed of 5x32bit words. The next CPMEM entry starts at the next 8x32bit words (0x0, 0x20,0x40, etc.).

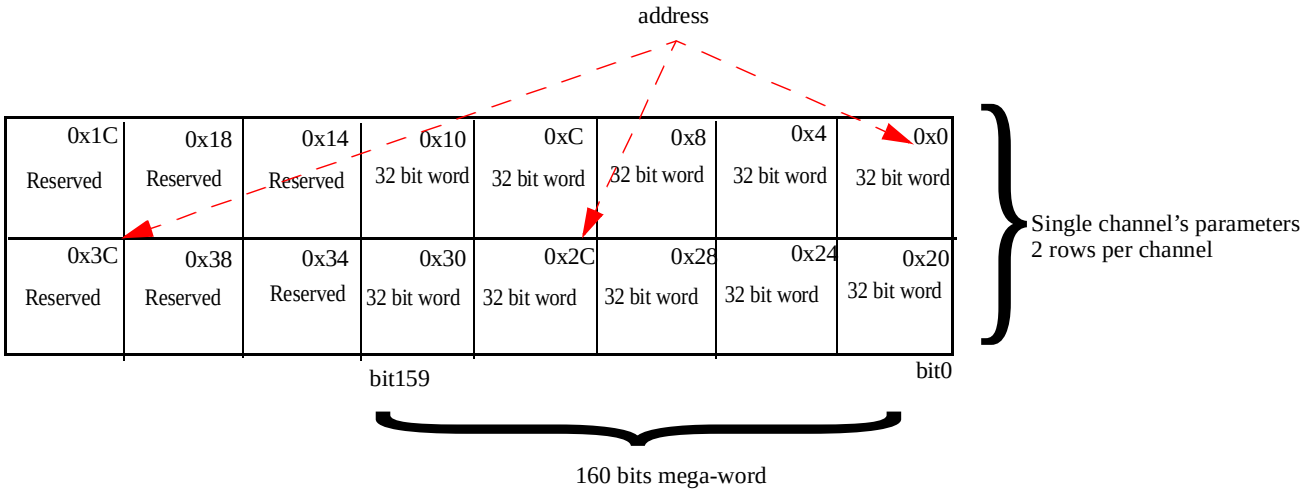


Figure 42-416. CPMEM's word structure

42.3.2.10.4 Alternate IDMAC settings

Some of the IDMAC channels support alternate flow. This means that a physical IDMAC channel can use alternate set of parameters. Switching between the flows is controlled by the CM. For the primary flow the IDMAC's settings are read from the channel's corresponding entry in the CPMEM. The alternate settings can be stored in anther entry on the CPMEM. The pointer to the alternate entry on the CPMEM is stored on the physical channel's corresponding IDMAC_SUB_ADDR parameters.

42.3.2.11 IDMAC's modes of operation

42.3.2.11.1 Rotation modes

Rotation is performed by the IDMAC and the Rotation unit inside the IC. The frame is partitioned into 8X8 pixels blocks. The IC reorders the pixels within a block. The IDMAC reorders the block. The reordering is done according to the ROT, VF & HF parameters in the CPMEM. The IC uses blocks of 8X8 pixels. As a result the AXI burst size is short. Knowing that it is more efficient to use longer bursts, the IDMAC provide the lock feature. This feature manipulates the arbitration mechanism such that a sequence of short bursts will be performed. The sequence is stopped once 8 accesses had been performed. Enabling the lock feature is done by setting the channel's corresponding IDMAC_LOCK_EN bit.

The following diagram illustrate various options for reordering of blocks.

- ROTATE means that the ROT bit is set

- HORIZONTAL means that the HF bit is set
- VERTICAL means that the VF bit is set

.

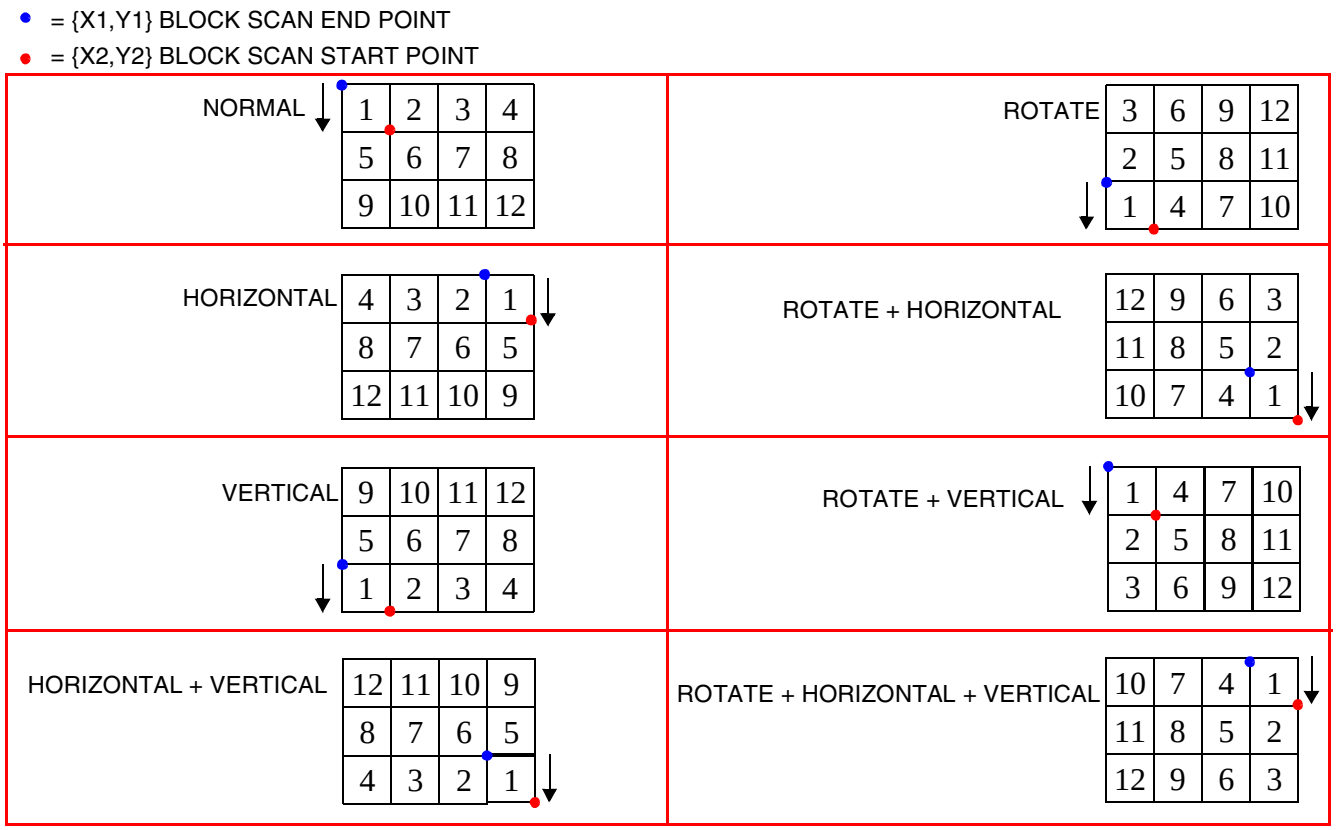


Figure 42-417. Rotation options

42.3.2.11.2 Frame size

The IPUv3EX supports various non-interleaved modes; the Frame Height (FH) and Frame Width (FW).

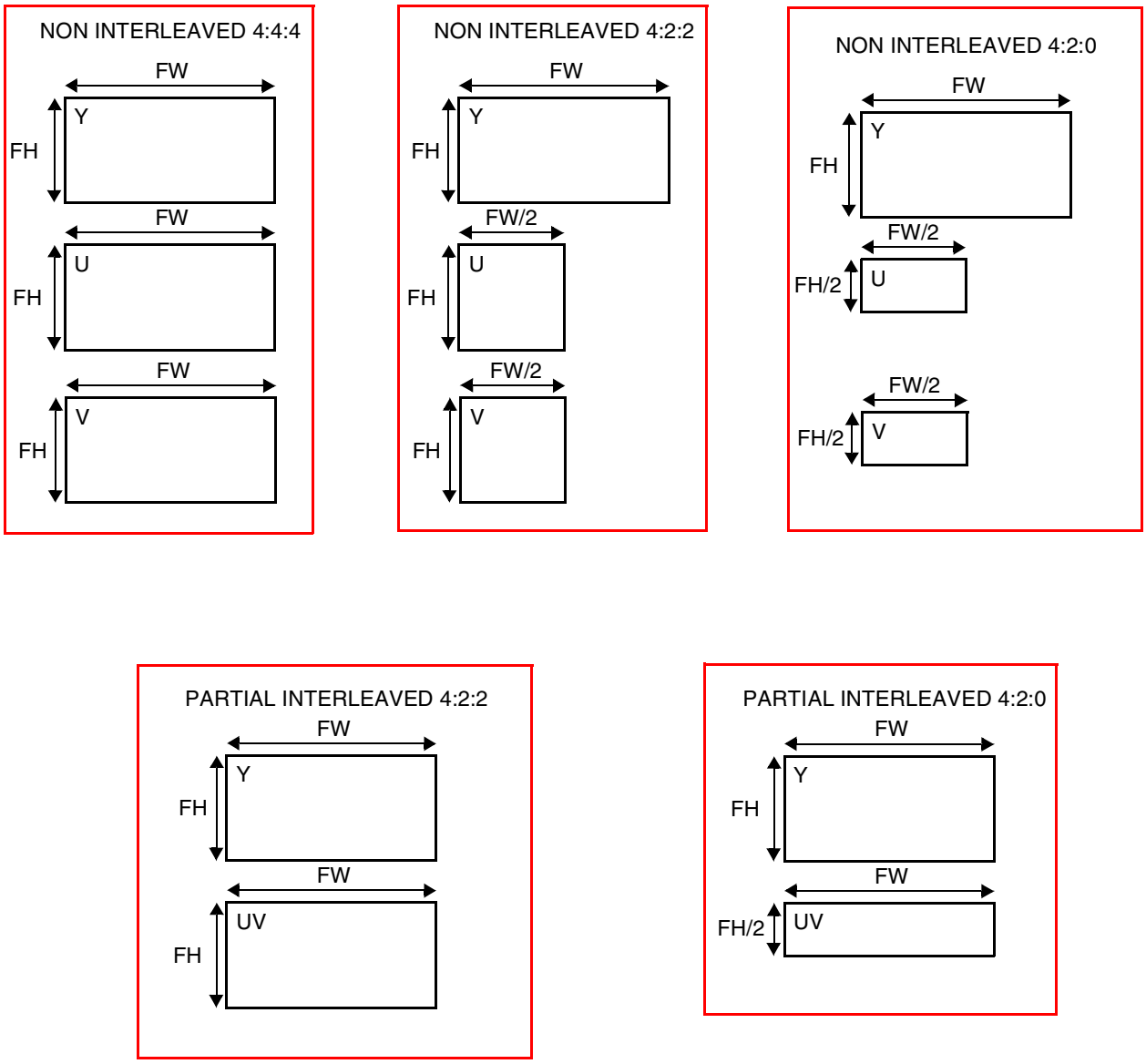


Figure 42-418. Frame size in various modes

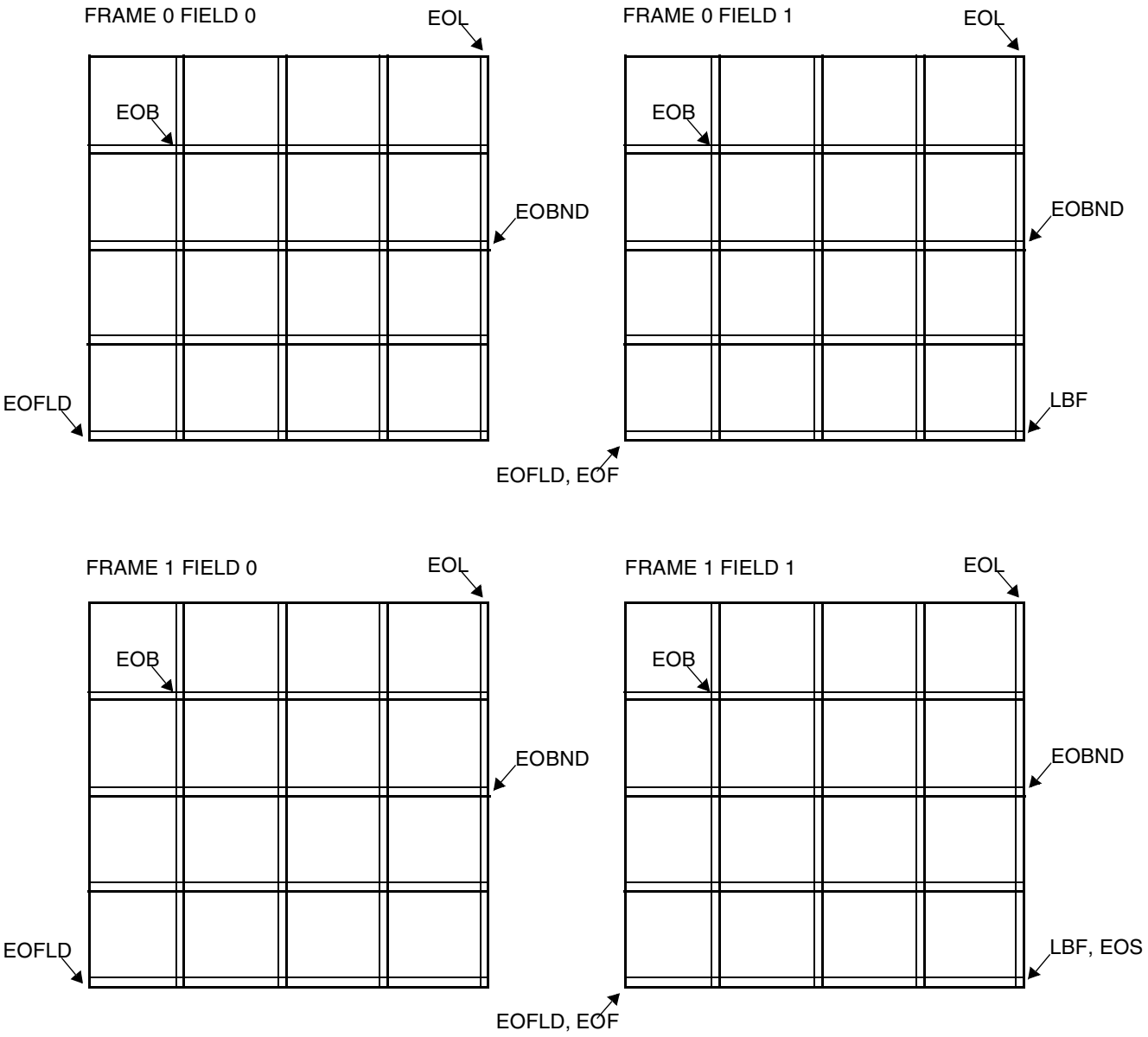


Figure 42-419. Frame's controls

42.3.2.12 IDMAC's restriction

The data must be received from the system's memory through the AXIR interface ("read" direction) "in-order" within a single burst. The entire burst can occur "out-of-order."

42.3.2.13 IDMAC's Endianness support

Byte Endianness - only LE (little endian) is supported

Pixel Endianness - both LE & BE are supported, only for read direction (only 4 BPP case is meaningful, supported only for the “read“ direction)

42.3.2.14 IDMAC’s internal events

Some of the IDMAC’s internal signals can be used for monitor the progress of flows. These bits can be polled by software. Some of these bits can be used to trigger an interrupt or an SDMA event. Table 42-417 describes the available events and their meaning

Table 42-417. IDMAC’s internal events

IDMAC event’s type	Monitored on	Event’s meaning
end of frame	IDMAC_EOF	This is the channel’s end of frame indication. This indication is asserted once the entire frame was read/written via the IDMAC. This indication is normally used as an interrupt or SDMA event
new frame acknowledge	IDMAC_NFACK	This indication means that the IDMAC acknowledge the new frame’s request from the module. It can be used to track the starting point of a flow or a frame.
new frame before end of fram error	IDMAC_NFB4EOF_ERR	This error indication may indicate on data lost. This indication is asserted when a new frame starts before the completion of the previous frame. For example if a real time input (from camera) was not written properly to the memory due to FIFO full condition.
end of scroll	IDMAC_EOS	end of scroll; This indication is asserted when the scroll counter finished counting its pre defined value
end of band	IDMAC_EOBND	This is the end of band indication. Any time IDMAC complete reading/writing a band it will assert this indication. This is useful to manually control a flow via channels working in band mode.
treshold	IDMAC_TH	Threshold crossing indication. The treshold is defined according to the TH parameter in the IDMAC.
channel busy	IDMAC_CH_BUSY	This signal is asserted when a channel is between NFACK event to EOF event. Negation of these indications is one of the conditions for low power modes handshake.

42.3.3 Camera Sensor Interface (CSI)

IPUv3EX has 2 identical camera sensor interfaces (CSI). The CSI description below refers to a single CSI.

42.3.3.1 CSI Block Diagram

The CSI Block Diagram is shown in Figure 42-420.

Figure 42-420. CSI Block Diagram

The CSI consists of synchronizer, interface logic, Data packing unit and Sensor Interface Control. The CSI is controlled via the peripheral bus registers. All programming parameters for the CSI are double buffered with synchronous change at the frame start.

The CSI gets data from the sensor, synchronizes the data and the control signals to the IPUv3EX clock (HSP_CLK), and transfer it according to configuration of DATA_DEST register to one or more of the following: IC, SMFC.

42.3.3.2 CSI Interface

CSI supports two types of interfaces. The interface is determined via the DATA_SOURCE register.

42.3.3.2.1 Parallel interface

In parallel interface a single value arrives in each clock, except when working in BT.1120 mode, in which two values arrive in each cycle. Each value can be 8-16 bit wide according to configuration of DATA_WIDTH. If DATA_WIDTH is configured to N, then 20-N LSB bits are ignored.

CSI can work with several data formats according to SENS_DATA_FORMAT configuration. In case the data format is YUV, the output of the CSI is always YUV444 (even if the data arrives in YUV422 format).

The polarity of the inputs can be configured using the registers SENS_PIX_CLK_POL, DATA_POL, HSYNC_POL and VSYNC_POL.

42.3.3.3 TEST MODE

When TEST_GEN_MODE register is configured to 1, the TEST MODE which is a debugging mode, is operated. The CSI generates the frame by itself and sends it to one of the destination units. The sent frame is a chess board composed of black and configured color squares. The configured color is set with the registers PG_B_VALUE, PG_G_VALUE and PG_R_VALUE. The data can be sent in different frequencies according to the configuration of DIV_RATIO register.

42.3.3.4 Sensor Image Frame Relations

Figure 42-421 illustrates the generalized relations between image frames produced by a sensor and accepted by the CSI. Generally, four frame definitions exist. The virtual frame A starts with the VSYNC signal. After vertical blanking starts frame B. The HSYNC signal indicates boundaries of the frame B. The frame B includes both the active sensor frame C and horizontal blanking intervals. A size of the blanking intervals depends on sensor type and programming. The size of the frame sent by the sensor (the actual pixels) has to be configured in the registers SENS_FRM_WIDTH and SENS_FRM_HEIGHT. The CSI selects a window (the frame D) inside the frame C by skipping rows and columns according to parameters

defined in registers HSC, VSC, ACT_FRM_HEIGHT and ACT_FRM_WIDTH. frame D is sent to the rest of the IPUv3EX.

Note: the following limitation must exist:

SENS_FRM_HEIGHT >= VSC + ACT_FRM_HEIGHT

SENS_FRM_WIDTH >= HSC + ACT_FRM_WIDTH

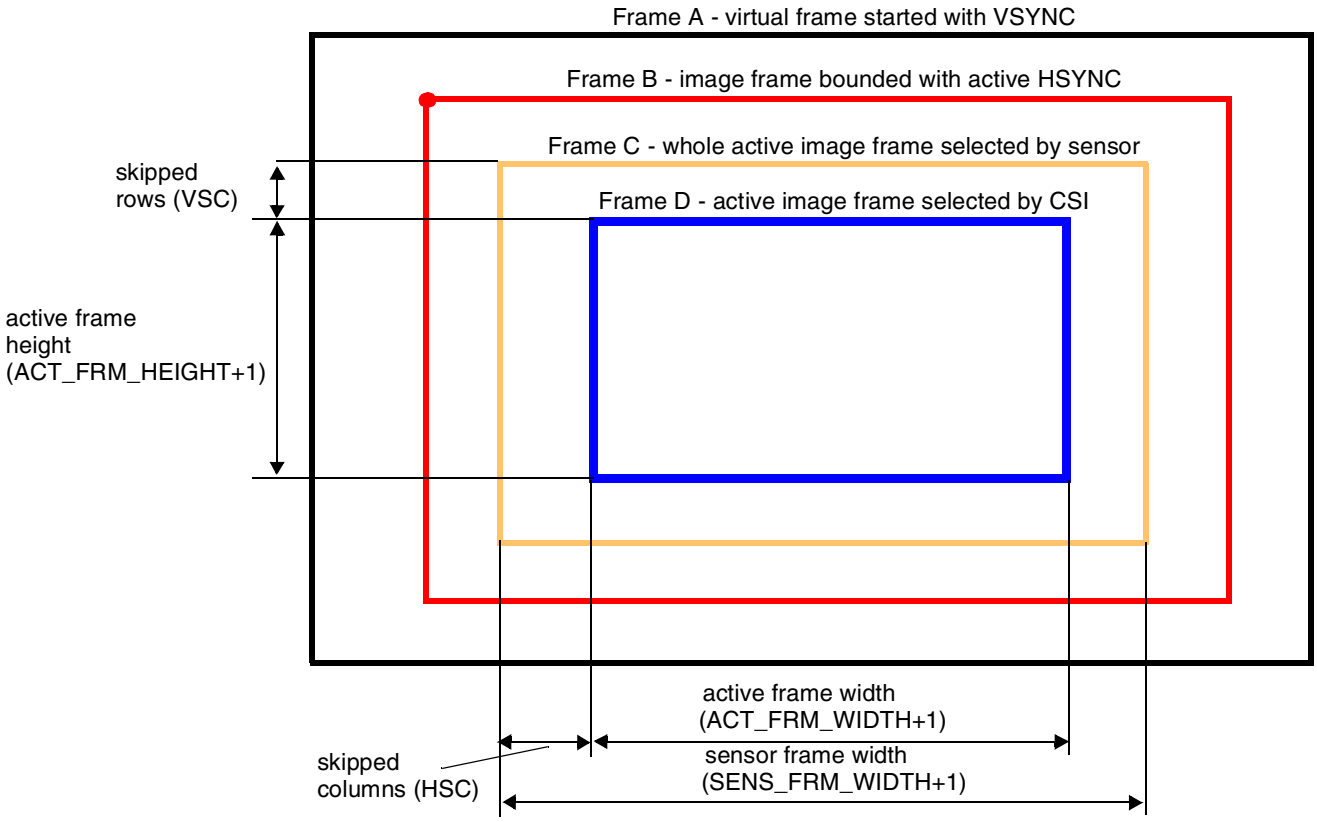


Figure 42-421. Sensor Image Frames

42.3.3.5 Companding

Companding is performed as follows:

input: 10-bit unsigned number.In case that component's size is bigger than 10 bit, 10 MSB bits are taken. In case that component's size is less than 10 bit, color extraction is performed.

First step: shift:

$$x \rightarrow clip(x + offset, 0, 1023) \qquad \textit{Eqn. 42-2}$$

where the offset is a 10 bit signed number that is configured in CPD_OFFSET1 and CPD_OFFSET2 registers.

second step - piecewise-linear map:

$$y = Min[255, (y1[k]+(((x-x1[k])*slope[k])>>6 + 1))>>1] \quad \textit{Eqn. 42-3}$$

where:

- the input range 0..1023 is partitioned to 16 equal segments: x1[k] .. x1[k+1] - 1 where x1[k] = 0, 64, 128, 192, 256, .. . 960.

- The linear map, in each segment, is characterized by y1[k] (9-bit unsigned number) and slope[k] (8-bit unsigned number).

Each destination unit can get the data after being companded depending on the configuration of CPD register. If this register is configured to 3'h0, the compander units are disable in order to save power. Parameters of the companding are equal for all destinations and can be set in the CPD registers.

2 companding units are located in the CSI. This is because when working in BT.1120 modes 2 components arrive in each cycle and the companding for each 2 components has to be in parallel. When CSI works in other mode, only one compander is needed, so the other one is disabled to save power.

42.3.3.6 Timing/Data mode protocols

CSI can work in several timing/data mode protocols according to SENS_PRTCL configuration:

42.3.3.6.1 gated mode

In this mode VSYNC is used to indicate beginning of a frame, HSYNC is used to indicate beginning of a raw. Sensor clock is ticking all the time.

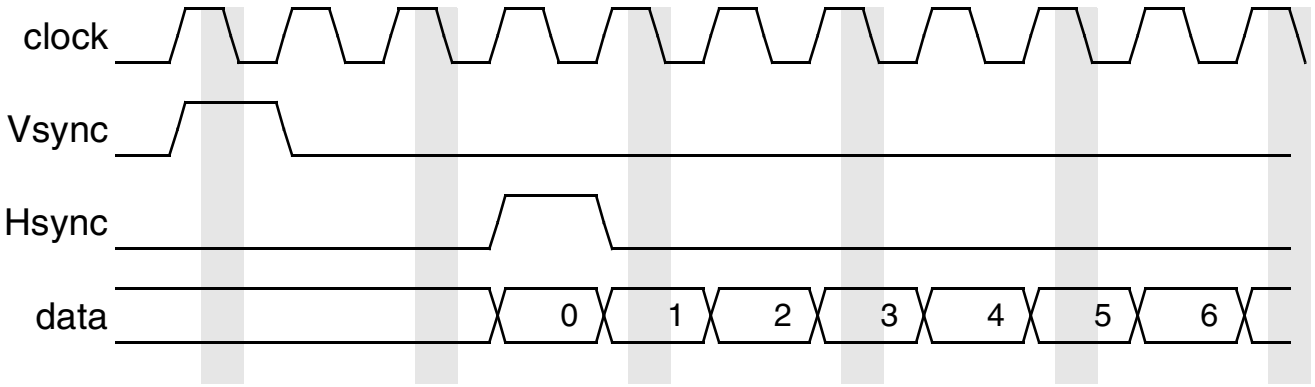


Figure 42-422. gated mode

42.3.3.6.2 non-gated mode

In this mode VSYNC is used to indicate beginning of a frame. Sensor clock is ticking only when data is valid. HSYNC is not used.

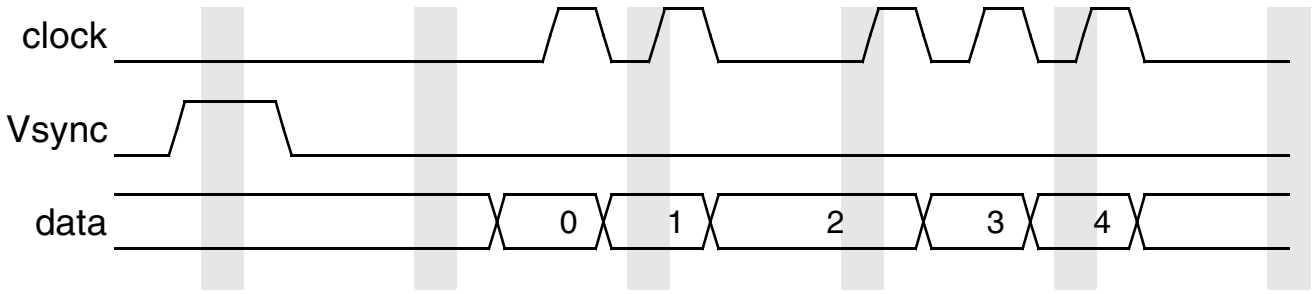


Figure 42-423. non-gated mode

42.3.3.6.3 BT.656 mode

In this mode the CSI works in compliance with recommendation ITU-R BT.656. The timing reference signals (frame start, frame end, line start, line end) are embedded in the data bus input. Each timing reference signal consists of a four word sequence. The first three words are fixed and configured in the CCIR_PRECOM register. The fourth word contains information defining field, the state of field blanking and the state of line blanking. these states are configured in registers CCIR_CODE_1 (for field 0) and CCIR_CODE_2 (for field 1).

In this mode in each cycle one value of data arrives

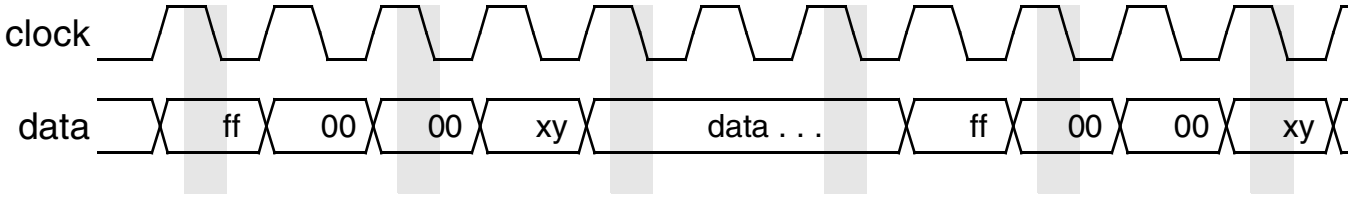


Figure 42-424. BT.656 mode

In this diagram the first three words are 0xff, 0x00, 0x00. The fourth word is XY and it includes the timing reference.

42.3.3.6.4 BT.1120 mode

In this mode the CSI works in compliance with recommendation ITU-R BT.1120. The timing reference signals (frame start, frame end, line start, line end) are embedded in the data bus input. Each timing reference signal consists of a four word sequence. The first three words are fixed and configured in the CCIR_PRECOM register. The fourth word contains information defining field, the state of field blanking and the state of line blanking. these states are configured in registers CCIR_CODE_1 (for field 0) and CCIR_CODE_2 (for field 1).

In this mode, the CSI can also work in DDR mode - data arrives on every edge of the clock. In addition, in each cycle two value of data arrive.

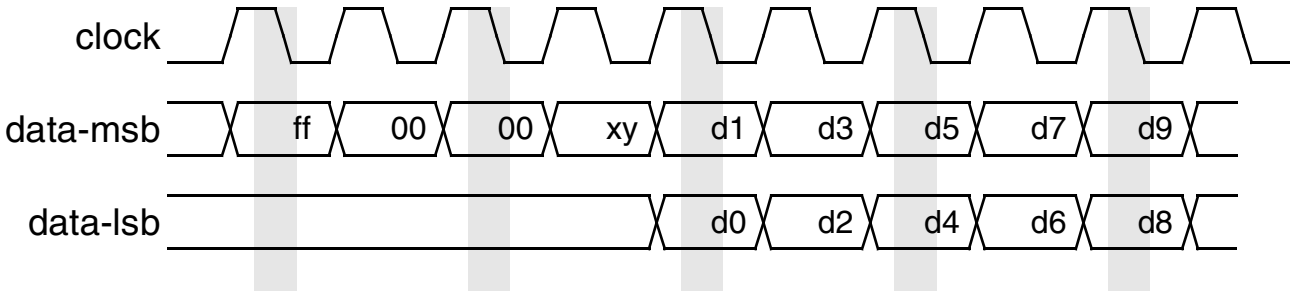


Figure 42-425. BT.1120 mode - SDR mode

In the above diagram each data arrives with the posedge of the clock.

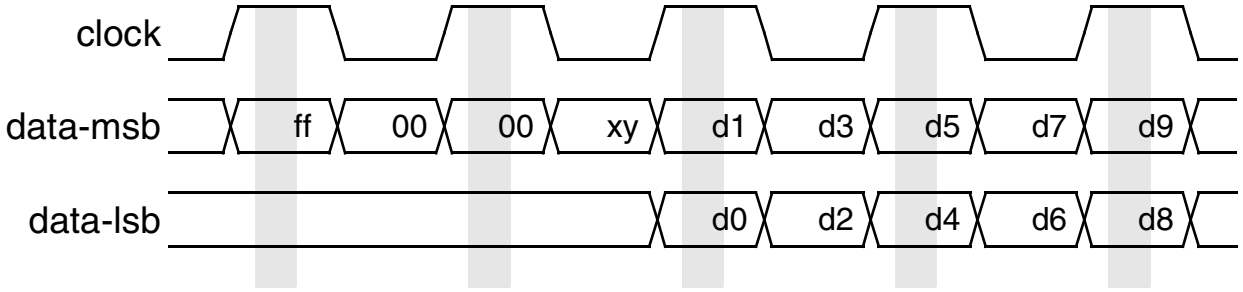


Figure 42-426. BT.1120 mode - DDR mode

In the above diagram data arrives in the posedge and the negedge of the clock.

42.3.3.7 packing to memory

the data bus output to the SMFC is 128-bit. The following table shows the how the CSI performs the packing before sending the data to the SMFC.

Table 42-418. Packing Unit

data format	component size	companded	regular packing	tight packing
Bayer, Generic data, JPEG	8	{8DC[1],8DC,8DC,...,8DC}	{8D,8D,8D,...,8D[2]}	NA
	9-16	{8DC,8DC,8DC,...,8DC}	{16DE[3],16DE,...,16DE}	NA
RGB, YUV	8	{8DC,8DC,8DC,8R}	{8D,8D,8D,8R}	NA
	9-10	{8DC,8DC,8DC,8R}	{16DE,16DE,16DE,16R[4]}	{10DE,10DE,10DE,2R}
	11-16	{8DC,8DC,8DC,8R}	{16DE,16DE,16DE,16R}	{10DT[5],10DT,10DT,2R}

[1] DC - data after being companded

[2] D - data arrived from sensor.

[3] DE - data after being extended.

[4] R- reserved bits

[5] DT - data after being truncated.

The tight packing functionality is enabled when PACK_TIGHT register is set. It is only used when data format is RGB or YUV and the data width is bigger than 8.

42.3.3.8 Skipping frames

Some of the frames that are sent to the SMFC can be skipped. skipped frames are ignore by the CSI and are not sent to the corresponding unit. . Using SKIP_SMFC and MAX_RATIO_SKIP_SMFC registers the user can define the frames for the SMFC that will be skipped.

42.3.3.9 CSI Restrictions

The frequency of the sensor clock must not be greater than the IPUv3EX clock (HSP_CLK)

SENS_FRM_HEIGHT >= VSC + ACT_FRM_HEIGHT

SENS_FRM_WIDTH >= HSC + ACT_FRM_WIDTH

42.3.4 Sensor Multi FIFO Controller (SMFC)

The Sensor Multifile Controller used as buffer between CSI and IDMAC. Two masters (CSIs) can be connected to SMFC. Both masters can be active simultaneously. Each master can send up to 4 frames, distinguished by csi_id bus. The frame can be mapped to one of four IDMAC channels via SMFC mapping registers. Each DMA channel have dedicated FIFO.

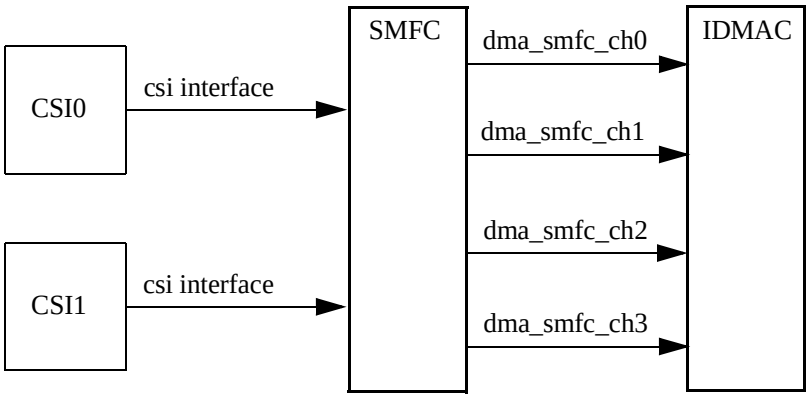


Figure 42-427. SMFC data flow

42.3.4.1 SMFC's Features

- Support two CSI masters and four DMA channels
- Automatic FIFO size calculation

42.3.4.2 SMFC's Functional description

SMFC support up two four DMA channels. Each channel have dedicated FIFO controller, as shown in Figure 42-428 Sampled data and frame ID are kept in the buffers till the buffer is selected by Round Robin Priority Mechanism. Then content of ID buffer are compared to CH#_MAP bits and corresponding FIFO controller is activated. As result a content of the buffer are copied to the RAM. The wptr and the "base" are used to calculate the location in the RAM. The rptr are following after wptr during dma_active signal, initiated by DMA.

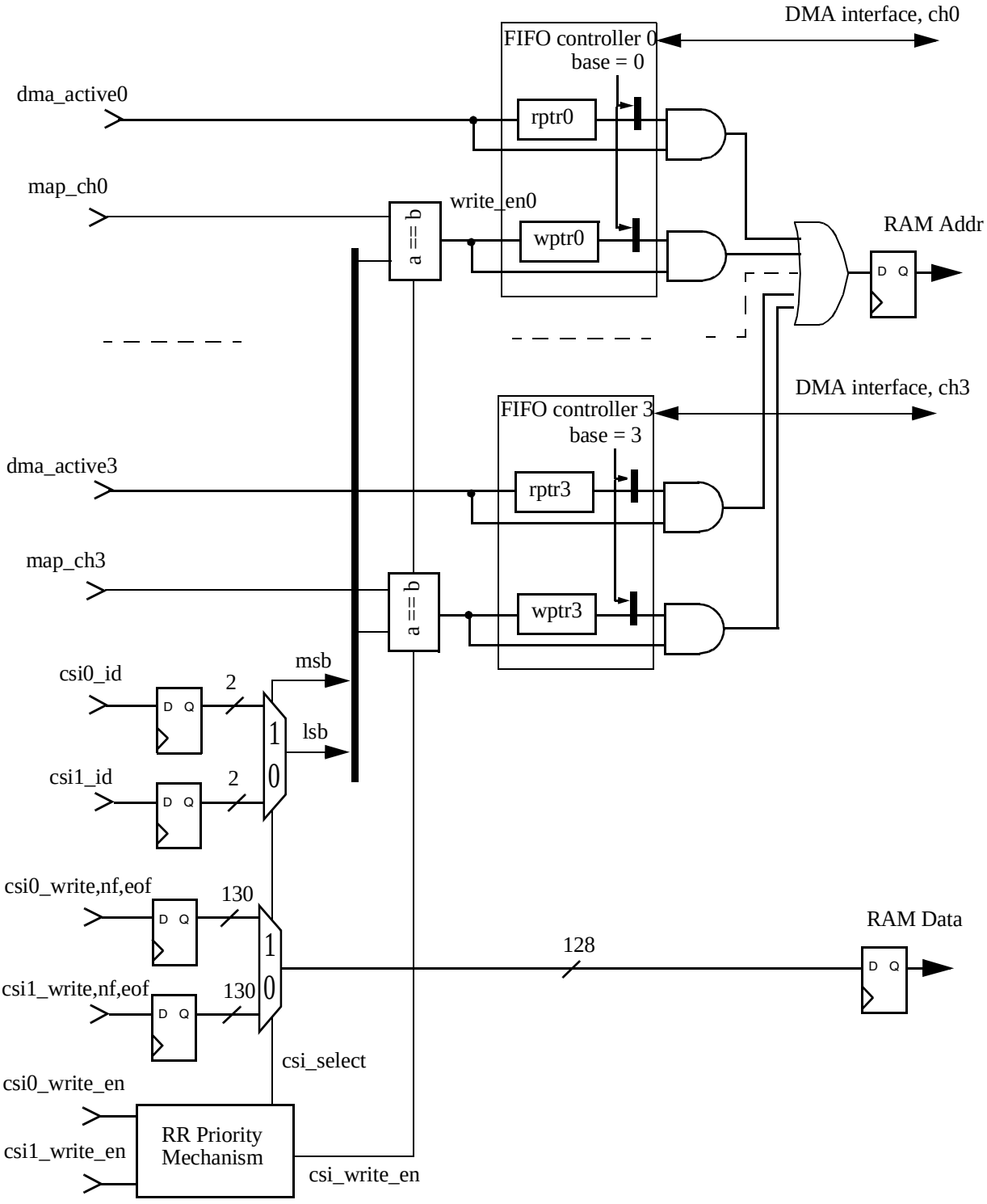


Figure 42-428. Sensor Multi FIFO Controller

Four FIFOs are implemented with single RAM due to the fact that the channels can't be active simultaneously. The memory space of SMFC divided to four equal sectors. Each FIFO have fixed base address - "base". The "base" used as MSB for corresponding pointer in order to calculate absolute address of the RAM. FIFO size of channels 1 and 3 are fixed and is equal to size of one sector. FIFO size of channels 0 and 2 depend from other channels as shown in Table 42-419. Other configuration are not allowed.

Table 42-419.

Number of DMA channels required	enable/disable of channels 3,2,1,0	FIFO size (sectors), per channels 3,2,1,0
1	0 0 0 1	0 0 0 4
2	0 1 0 1	0 2 0 2
3	1 1 0 1	1 1 0 2
4	1 1 1 1	1 1 1 1

NOTE: channels should not be enabled after activation of channel 0 or/and channel 2. This can cause to overlapping of FIFO areas and other malfunctions.

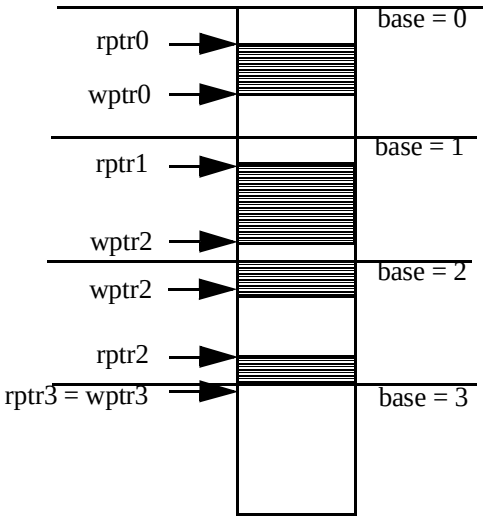


Figure 42-429. SMFC memory map when DMA channels 3,2,1,0 are enabled

42.3.4.2.1 SMFC Master interface.

SMFC Master interface shown in Figure 42-430

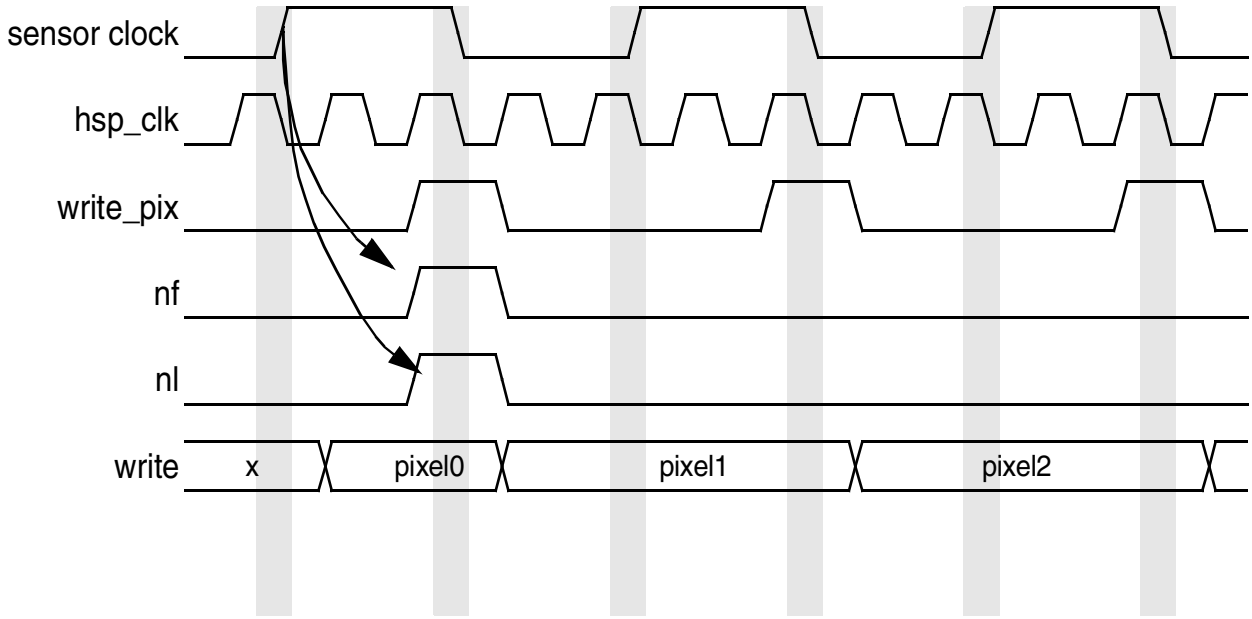


Figure 42-430. Timing diagram of SMFC slave interface

Sensor clock can be asynchronous to IPUv3EX clock (hsp_clk). The CSI synchronize data arrived from the sensor to hsp_clk. The csi_write signal indicate when data on csi_pix bus can be sampled by hsp_clk clock.

42.3.4.2.2 Restrictions

1. DMA channels should not be enabled after activation of channel 0 or/and channel 2. This can cause to overlapping of FIFO area and other malfunctions.
2. Watermark set value should be greater than watermark clear value.
3. One frame should not be mapped to few DMA channel.

42.3.5 Image Converter (IC)

42.3.5.1 IC Block Diagram

The IC Block Diagram is shown in Figure 42-431.

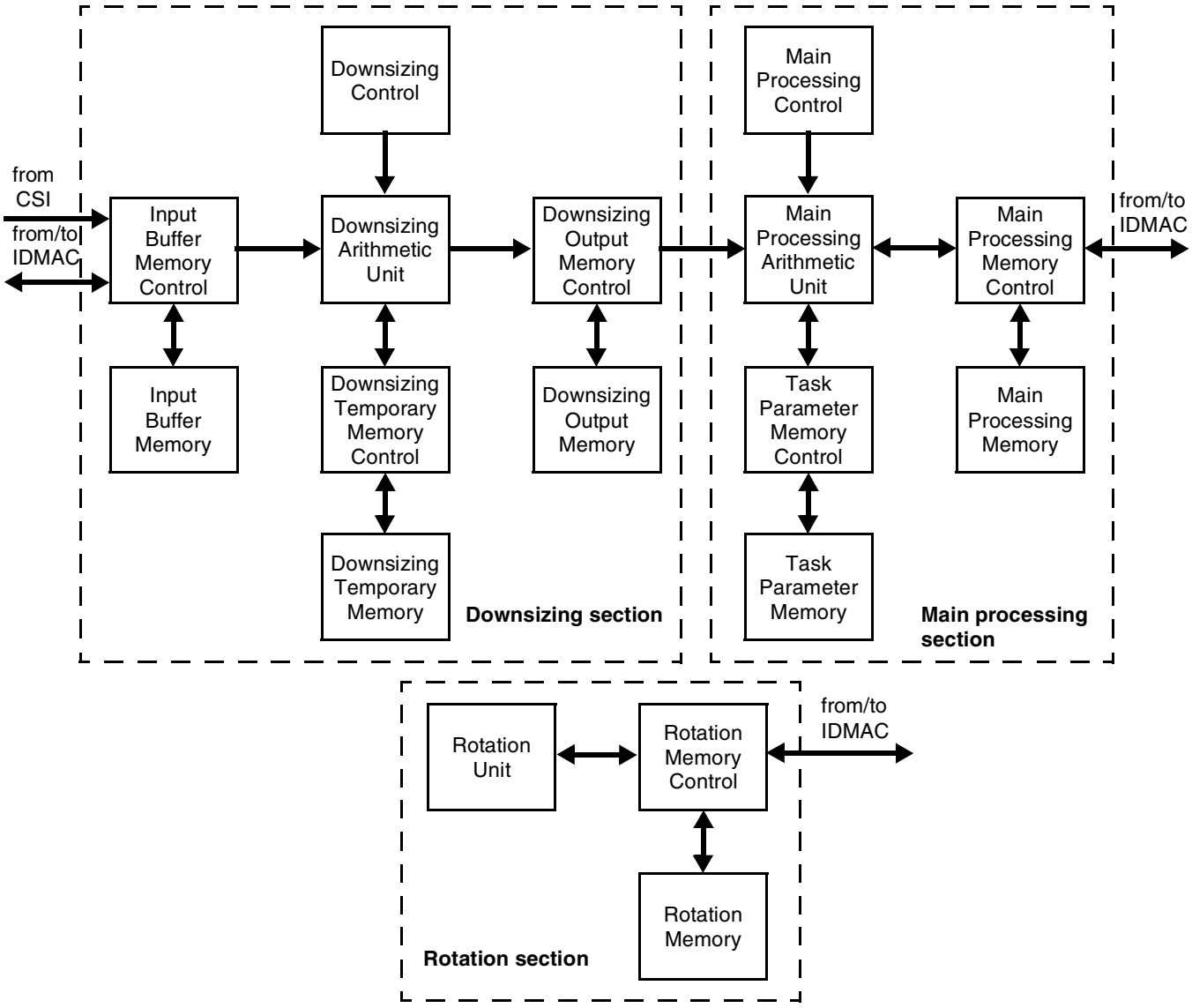


Figure 42-431. IC Block Diagram

The IC contains three processing sections: downsizing, main processing and rotation. The module is controlled via the peripheral bus registers. Some processing parameters should be written by the MCU to the Task Parameter Memory. Writing to the memory is performed via the AHB bus.

42.3.5.2 Processing tasks

Each of three processing section performs up to three processing tasks with time sharing:

1. Preprocessing task for encoding.
2. Preprocessing task for displaying image from sensor (viewfinder).
3. Postprocessing task.

The tasks are performed by single hardware. The MCU configures each task before enabling it. Tasks switching is transparent for the MCU. The time unit for task switching in the downsizing section corresponds to a processing time of one burst of eight pixels, in the main processing section - to a processing time of one image line, in the rotation section - to a processing time of one image frame.

All three tasks includes similar operations controlled by commands. Task configuring consists in definition of commands for each task as described in Table 42-420.

Table 42-420. Task Commands

Command code	Command	Processing unit	Command parameters	Description
EN	Task Enable	DSU, MPU		Task will enabled from next frame. Task 1 is preprocessing for encoding. Task 2 is preprocessing for viewfinder. Task 3 is postprocessing.
	Downsizing	DSU	Downsizing ratio (GCR)	Downsizing ratio 1:1, 2:1, 4:1
	Resizing	MPU	Resizing ratio (GCR)	Resizing ratio from 2:1 to 1:M Resizing ratio = N:M; M =2^13; N = floor(M*(SI-1)/(SO-1)); SI - input size; SO - output size
CSC1	Color space conversion 1	MPU	Color conversion coefficients and offsets (TPM)	Color conversion matrix 1
CSC2	Color space conversion 2	MPU	Color conversion coefficients and offsets (TPM)	Color conversion matrix 2 Used only for Task 2 and Task 3 It is not performed if the first color space conversion or combining are disabled
GLOB_A	Global alpha	MPU		Used only for Task 2 and Task 3
CMB	Combining	MPU		Combining video with graphics. Used only for Task2 and Task3.

The MCU writes the commands to the IC_CONF Register. Because there is no double buffering for all the IC parameters, the MCU must disable a task before changing its parameters. After being disabled, the task is still allowed to complete its current frame execution. At frame finish, the IPUv3EX sends an interrupt to the MCU indicating that the MCU can change task parameters. The MCU enables the task again and task execution is resumed from start of the next frame.

42.3.5.3 Downsizing Section

The sensor data from the CSI is written into a FIFO located in the Input Buffer Memory. Depending on programmed processing flow, the FIFO data can be sent to the system memory via the IDMAC or straight forward to the Downsizing Unit. In the first case, the data is processed by the MCU and returned by the IDMAC to another FIFO located in the Input Buffer Memory.

For postprocessing, the IDMAC transfers a data from the system memory to the third FIFO. The data is read by the Downsizing Unit when the postprocessing is performed.

Each or three FIFOs has 128 pages. Each page can store one burst of 2 or 4 words which corresponds to 8 or 16 pixels. The size of the burst is defined according to the channel's corresponding CB#_BURST_SIZE bit. The memory word width is 128 bits. Each memory word contains color components of four adjacent pixels or 16 bytes of generic data (e.g. Bayer). Access to the FIFOs is controlled by the Input Buffer Memory Control.

The Downsizing Unit performs averaging and decimation of image pixels both in horizontal and vertical directions according to the following equations:

$$HP_{R,c} = \frac{1}{DS\_R\_H} \sum_{k=0}^{DS\_R\_H-1} IP_{r+k,c}$$

$$VP_{R,C} = \frac{1}{DS\_R\_V} \sum_{l=0}^{DS\_R\_V-1} HP_{R,c+l}$$

where $IP_{r,c}$ - the input pixel, $HP_{R,c}$ - the pixel after horizontal downsizing, $VP_{R,C}$ - the pixel after vertical downsizing, DS_R_H and DS_R_V - the horizontal and vertical downsizing ratios according to the IC_PRP_ENC_RSC, IC_PRP_VF_RSC and IC_PP_RSC Registers. The final calculation result is rounded to 8 bits.

Each of three downsizing tasks processes the data by bursts of 8 pixels. Normally, the current task runs until emptying the corresponding input FIFO. After finishing burst processing, the Downsizing Unit may switch between the current task and another task with higher priority, if the Input Buffer Memory has received a burst for this new task.

Averaging is performed firstly in the horizontal direction. All color components of a pixel are processed in parallel. After horizontal averaging has finished for a single output pixel, the new pixel value is added to the corresponding pixel value of a temporary row derived from previous averaging steps. This provides vertical averaging of the pixels. The temporary row is stored in the Downsizing Temporary Memory. The memory word width matches one accumulated pixel width (36 bits). There are three temporary rows stored in this memory - one per downsizing task.

After vertical averaging has been finished, the output row is written to the Downsizing Output Memory. The memory word of 48 bits includes two output pixels. For each task, the memory has a double buffer of one row. When the Downsizing Unit fills the foreground part of the double buffer, the Main Processing

Unit takes pair or pixels from the background part. After the new downsized row has been ready, the foreground and background memory pointers are swapped.

42.3.5.4 Main Processing Section

The Main Processing Unit reads pairs of pixels from the Downsizing Output Memory background part. It processes the complete pixel row for the current task and after that switches to another task if the input data for this new task is ready. For each task, the Main Processing Unit is able to perform the following sequence of operations:

1. Horizontal flipping the image (optional) performed with reading from the Downsizing Output Memory. Flipping is enabled via the VF, HF & ROT parameters of the corresponding DMA channels (See: Table 42-415. Channel Parameters Memory for non-interleaved and See: Table 42-416. Channel Parameters Memory for interleaved) responsible for output of the task results. The preprocessing task for encoding uses the VF, HF & ROT parameters from IDMAC channel #20, the preprocessing task for viewfinder - from the IDMAC channel #21, the postprocessing task - from the IDMAC channel #22.
2. Horizontal resizing by bilinear interpolation between two adjacent pixels received from the Downsizing Output Memory according to the equation:

$$HP_{R,c} = IP_{r,c} + RS\_C\_H \cdot (IP_{r+1,c} - IP_{r,c})$$

where RS_C_H - the current horizontal resizing coefficient. The calculation result is rounded to 8 bits. The resizing coefficient is calculated as

$$RS\_C\_H = \left(\sum_{k=0}^{R-1} RS\_R\_H \right) \bmod (8196)$$

where RS_R_H - the horizontal resizing ratio from the IC_PRP_ENC_RSC, IC_PRP_VF_RSC and IC_PP_RSC Registers. The RS_R_H parameter is equal to a numerator N of the resizing ratio N:M with M =2^13.

The resulting row of the horizontal resizing is stored in the Task Parameter Memory.

3. Vertical resizing by bilinear interpolation between the current and previous results of horizontal resizing. Both current and previous results of horizontal resizing is stored in the Task Parameter Memory. Resizing is accomplished according to the equation:

$$VP_{R,C} = HP_{R,c} + RS\_C\_V \cdot (HP_{R,c+1} - HP_{R,c})$$

where RS_C_V - the current vertical resizing coefficient. The calculation result is rounded to 8 bits. The

resizing coefficient is calculated as

$$RS\_C\_V = \left(\sum_{k=0}^{C-1} RS\_R\_V \right) \bmod (8196)$$

where RS_R_V - the horizontal resizing ratio from the IC_PRP_ENC_RSC, IC_PRP_VF_RSC and IC_PP_RSC Registers. The RS_R_V parameter is equal to a numerator N of the resizing ratio N:M with M =2^13.

At completion of vertical resizing, this row is updated - the current result of horizontal resizing replaces the previous one.

4. First color space conversion YUV to RGB or RGB to YUV with the conversion matrix CSC1. The conversion matrix coefficients are programmable. They are stored in the Task Parameter Memory. The conversion equations are:

$$\begin{aligned} Z_0 &= 2^{SCALE-1} \cdot (X_0 \cdot C_{00} + X_1 \cdot C_{01} + X_2 \cdot C_{02} + A_0) \\ Z_1 &= 2^{SCALE-1} \cdot (X_0 \cdot C_{10} + X_1 \cdot C_{11} + X_2 \cdot C_{12} + A_1) \\ Z_2 &= 2^{SCALE-1} \cdot (X_0 \cdot C_{20} + X_1 \cdot C_{21} + X_2 \cdot C_{22} + A_2) \end{aligned}$$

where for YUV to RGB: X_0=Y, X_1=U, X_2=V, Z_0=R, Z_1=G, Z_2=B,
for RGB to YUV: X_0=R, X_1=G, X_2=B, Z_0=Y, Z_1=U, Z_2=V.

All the parameters of the conversion matrix are written by the MCU to the Task Parameter Memory (See: 42.3.5.6 IC Task Parameter Memory). The final calculation result is limited according to the SAT_MODE parameter and rounded to 8 bits.

5. Combining video with graphics. There are the following combining options:
   - local alpha blending,
   - global alpha blending,
   - use of key color.

   If both alpha blending and color keying are enabled, color keying has higher priority (graphic pixels of the key color are fully transparent independently on the alpha value).
   Combining mode is selected via the IC_CONF Register. The combining equation is:

$$OP = IGP \cdot \alpha + IVP \cdot (1 - \alpha)$$

where IGP - an input graphics pixel, IVP - an input video pixel, α=(A+floor(A/128))/256 - an alpha value, A - a global or local transparency parameter. The global A is written in the IC_CMBP_1 Register, the local A arrives together with the graphics pixel.

A graphics pixel becomes transparent when color keying is enabled and a pixel color matches a key color (independently on an alpha parameter).

The graphics data is read from a FIFO located in the Main Processing Memory. The FIFO contains 32 pages of size of 8 pixels. The size of the burst is defined according to the channel's corresponding CB#_BURST_SIZE bit. The graphics pixel format in the FIFO is RGB or RGBA or YUV 4:4:4 or YUVA. The graphics data is loaded by the IDMAC to the FIFO from the system memory.

6. Second color space conversion YUV to RGB or RGB to YUV with the conversion matrix CSC2. Typically this is color space conversion RGB to YUV. It is not performed if the first color space conversion or combining are disabled. This operation is designed for TV output. The conversion matrix coefficients are programmable with the same parameters and equation as the first color conversion.

All the operation are executed by an unified processing unit sequentially. Steps 1 and 2 cannot be interrupted by another task. All other steps can be interrupted by a task with higher priority if an input row is ready for this task. Preprocessing tasks priority is higher than postprocessing task priority.

The processing unit consists of three identical parts for each color component. All three color components are processed in parallel. Each of the processing operations can be enabled or disabled by an appropriate command according to.

The processing results are written to an output FIFO located in the Main Processing Output Memory row-by-row. The FIFO contains 32 pages, each page can include one burst of 8 or 16 pixels. The size of the burst is defined according to the channel's corresponding CB#_BURST_SIZE bit. The IDMAC transfers the output bursts to the system memory or to the display via DMFC (Channel 21only). The Main Processing Memory contains three buffers for each tasks: the temporary row buffer, the graphics FIFO and the output FIFO. Each memory word (128 bits) stores 4 adjacent pixels in formats RGB or RGBA or YUV 4:4:4.

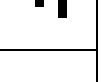
42.3.5.5 Rotation Section

The rotation section includes the Rotation Memory which stores an input rectangular block of 8x8 pixels and an output FIFO containing four pages of 8 pixels each one. The Rotation Memory word width corresponds to four adjacent pixels - 96 bits. The input block is loaded to the memory by the IDMAC like to a FIFO.

The Rotation Unit rewrites pixels from the input block to the output FIFO with corresponding relocation of a pixel inside the block. Rotation and/or left/right flipping and/or up/down flipping are enabled separately for each of three tasks. Configuring the rotation and flipping options is performed via the VF, HF & ROT parameters of the corresponding DMA channels (See: Table 42-415. Channel Parameters Memory for non-interleaved and See: Table 42-416. Channel Parameters Memory for interleaved) responsible for task data input. The preprocessing task for encoding uses the VF, HF & ROT parameters from the IDMAC channel #45, the preprocessing task for viewfinder - from the IDMAC channel #46, the postprocessing task - from the IDMAC channel #47.

Rotation and flip options are shown in Table 42-421.

Table 42-421. Rotation and Flip Options

ROT	FLR	FUD	Image
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

After finishing the rotation task, the IDMAC returns the output FIFO content to the system memory. When writing to the system memory, the IDMAC changes a location of the block relative to an input block location in order to provide proper rotation of the whole frame. Rotation tasks switching is performed after completion of rotation of the whole frame.

42.3.5.6 IC Task Parameter Memory

Table 42-422 presents IC task parameter memory details.

Table 42-422. IC Parameters

Address[1]	Word[2]	Parameter	Field	Description
x2008	Encoding CSC1 word0 and word1	C22	8:0	Coefficients of color conversion matrix1 for encoding task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx[3];
		C11	17:9	
		C00	26:18	
		A0	39:27	Offset of color conversion matrix1 for encoding task: Offset format is sxx.xxxxxxxxxx
		SCALE	41:40	Scale of coefficients for color conversion matrix1 for encoding task: 0 --> coefficients *2 1 --> coefficients*1 2 --> coefficients*0.5 3 --> coefficients*0.25
		SAT_MODE	42:42	Saturation mode for color conversion matrix1 for encoding task: 0 --> (min, max) = (0, 255) 1 --> (min, max) = (16, 240) for Z1,Z2 1 --> (min, max) = (16, 235) for Z0
x2010	Encoding CSC1 word2 and word3	C20	8:0	Coefficients of color conversion matrix1 for encoding task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C10	17:9	
		C01	26:18	
		A1	39:27	Offset of color conversion matrix1 for encoding task: Offset format is sxx.xxxxxxxxxx
x2018	Encoding CSC1 word4 and word5	C21	8:0	Coefficients of color conversion matrix1 for encoding task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C12	17:9	
		C02	26:18	
		A2	39:27	Offset of color conversion matrix1 for encoding task: Offset format is sxx.xxxxxxxxxx

Table 42-422. IC Parameters (continued)

Address[1]	Word[2]	Parameter	Field	Description
x4028	Viewfinder CSC1 word0 and word1	C22	8:0	Coefficients of color conversion matrix1 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C11	17:9	
		C00	26:18	
		A0	39:27	Offset of color conversion matrix1 for viewfinder task: Offset format is sxx.xxxxxxxxxx
		SCALE	41:40	Scale of coefficients for color conversion matrix1 for viewfinder task: 0 -->coefficients *2 1--> coefficients*1 2--> coefficients*0.5 3-->coefficients*0.25
		SAT_MODE	42:42	Saturation mode for color conversion matrix1 for viewfinder task: 0 --> (min, max) = (0, 255) 1 --> (min, max) = (16, 240) for Z1,Z2 1 --> (min, max) = (16, 235) for Z0
x4030	Viewfinder CSC1 word2 and word3	C20	8:0	Coefficients of color conversion matrix1 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C10	17:9	
		C01	26:18	
		A1	39:27	Offset of color conversion matrix1 for viewfinder task: Offset format is sxx.xxxxxxxxxx
x4028	Viewfinder CSC1 word4 and word5	C21	8:0	Coefficients of color conversion matrix1 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C12	17:9	
		C02	26:18	
		A2	39:27	Offset of color conversion matrix1 for viewfinder task: Offset format is sxx.xxxxxxxxxx

Table 42-422. IC Parameters (continued)

Address[1]	Word[2]	Parameter	Field	Description
x4040	Viewfinder CSC2 word0 and word1	C22	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C11	17:9	
		C00	26:18	
		A0	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx
		SCALE	41:40	Scale of coefficients for color conversion matrix1 for viewfinder task: 0 -->coefficients *2 1--> coefficients*1 2--> coefficients*0.5 3-->coefficients*0.25
		SAT_MODE	42:42	Saturation mode for color conversion matrix2 for viewfinder task: 0 --> (min, max) = (0, 255) 1 --> (min, max) = (16, 240) for Z1,Z2 1 --> (min, max) = (16, 235) for Z0
x4048	Viewfinder CSC2 word2 and word3	C20	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C10	17:9	
		C01	26:18	
		A1	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx
x4050	Viewfinder CSC2 word4 and word5	C21	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C12	17:9	
		C02	26:18	
		A2	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx

Table 42-422. IC Parameters (continued)

Address[1]	Word[2]	Parameter	Field	Description
x6060	Postprocessing CSC1 word0 and word1	C22	8:0	Coefficients of color conversion matrix1 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C11	17:9	
		C00	26:18	
		A0	39:27	Offset of color conversion matrix1 for viewfinder task: Offset format is sxx.xxxxxxxxxx
		SCALE	41:40	Scale of coefficients for color conversion matrix1 for postprocessing task: 0 -->coefficients *2 1--> coefficients*1 2--> coefficients*0.5 3-->coefficients*0.25
		SAT_MODE	42:42	Saturation mode for color conversion matrix1 for postprocessing task: 0 --> (min, max) = (0, 255) 1 --> (min, max) = (16, 240) for Z1,Z2 1 --> (min, max) = (16, 235) for Z0
x6068	Postprocessing CSC1 word2 and word3	C20	8:0	Coefficients of color conversion matrix1 for postprocessing task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C10	17:9	
		C01	26:18	
		A1	39:27	Offset of color conversion matrix1 for post-processing task: Offset format is sxx.xxxxxxxxxx
x6070	Postprocessing CSC1 word4 and word5	C21	8:0	Coefficients of color conversion matrix1 for post-processing task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C12	17:9	
		C02	26:18	
		A2	39:27	Offset of color conversion matrix1 for postprocessing task: Offset format is sxx.xxxxxxxxxx

Table 42-422. IC Parameters (continued)

Address[1]	Word[2]	Parameter	Field	Description
x6078	Postprocessing CSC1 word0 and word1	C22	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C11	17:9	
		C00	26:18	
		A0	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx
		SCALE	41:40	Scale of coefficients for color conversion matrix1 for viewfinder task: 0 -->coefficients *2 1--> coefficients*1 2--> coefficients*0.5 3-->coefficients*0.25
		SAT_MODE	42:42	Saturation mode for color conversion matrix2 for viewfinder task: 0 --> (min, max) = (0, 255) 1 --> (min, max) = (16, 240) for Z1,Z2 1 --> (min, max) = (16, 235) for Z0
x6080	Postprocessing CSC1 word2 and word3	C20	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C10	17:9	
		C01	26:18	
		A1	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx
x6088	Postprocessing CSC1 word4 and word5	C21	8:0	Coefficients of color conversion matrix2 for viewfinder task: Z0 = X0*C00 + X1*C01 +X2*C02+A0; Z1 = X0*C10 + X1*C11 +X2*C12+A1; Z2 = X0*C20 + X1*C21 +X2*C22+A2; Coefficients format is s.xxxxxxxx;
		C12	17:9	
		C02	26:18	
		A2	39:27	Offset of color conversion matrix2 for viewfinder task: Offset format is sxx.xxxxxxxxxx

1 The address documented in this table is the relative address within the TPM. This is the address that should be accessed when writing or reading from this memory via the AHB bus.

2 Each word is aligned to 64 bit accessible via the AHB bus in 2 separate 32bit accesses.

3 s - sign position, x - binary digit position

42.3.5.7 IC's DMA channels

The table below mas the IDMAC channels of the IC and the IRT to the corresponding tasks. The IC's channel name is the name of the channel at IC level. The IC channel name is referred on the IC's programming model.

Table 42-423. IC's DMA Channels

IDMAC's channel number	IC's channel name	Read/Write	Source	Destination	Processing Flow Purpose
20	CB0	Write	IC ENC	Memory	Preprocessing data from IC (encoding task) to memory
21	CB1	Write	IC VF	Memory/DMFC	Preprocessing data from IC (viewfinder task) to memory; This channel can be configured to send the data directly to the DMFC. This is done by programming the ic_dmfc_sel bit.
22	CB2	Write	IC PP	Memory	Postprocessing data from IC to memory
14	CB3	Read	Memory	IC VF	Graphics data for combining (viewfinder task)
15	CB4	Read	Memory	IC PP	Graphics data for combining (post-processing task)
11	CB5	Read	Memory	IC PP	Postprocessing data from memory
12	CB6	Read	Memory	IC VF	Preprocessing data from sensor stored in memory (for example Bayer)
5	CB7	Write	IC	Memory	Direct data from IC (sensor data) to memory
48	CB8	Write	ENC ROT	Memory	Preprocessing data after rotation (encoding task)
49	CB9	Write	VF ROT	Memory	Preprocessing data after rotation (viewfinder task)
45	CB10	Read	Memory	ENC ROT	Preprocessing data for rotation (encoding task)
46	CB11	Read	Memory	VF ROT	Preprocessing data for rotation (viewfinder task)
50	CB12	Write	PP ROT	Memory	Postprocessing data after rotation
47	CB13	Read	Memory	PP ROT	Postprocessing data for rotation

42.3.5.8 IC restrictions

- The input's frame width to the IC must be a multiplication of 8 pixels
- When performing resizing the frame width must be multiple of burst size - 8 or 16 pixels as defined by CB#_BURST_16 parameter.

42.3.5.9 IC bridge

The IC module utilize a single memory to serve read and write channels. These memories are the IBM, RM and MPM. The IDMAC has separate mechanism to handle read and write channels. As a result a contention between read and write channels may occur on each one of the memories. In order to resolve the contention an IC bridge module is connected between the channels associated with this memory. The bridge prioritize a read channel over a write channel. In order to avoid starvation of the write channels, the

user can limit the maximum consecutive requests of the same channel that will be served. The limitation is done by programming the memory's corresponding <>_brdg_max_rq field.

42.3.6 Display Port

The display port handles all the IPUv3EX features targeted for controlling and sending data to the display. The display port consists of 4 modules.

DC - a display controller,

DP - a display processor,

DMFC - a display multi-FIFO controller

DI - a display interface. The DI is instantiated twice to provide two symmetrical display interfaces.

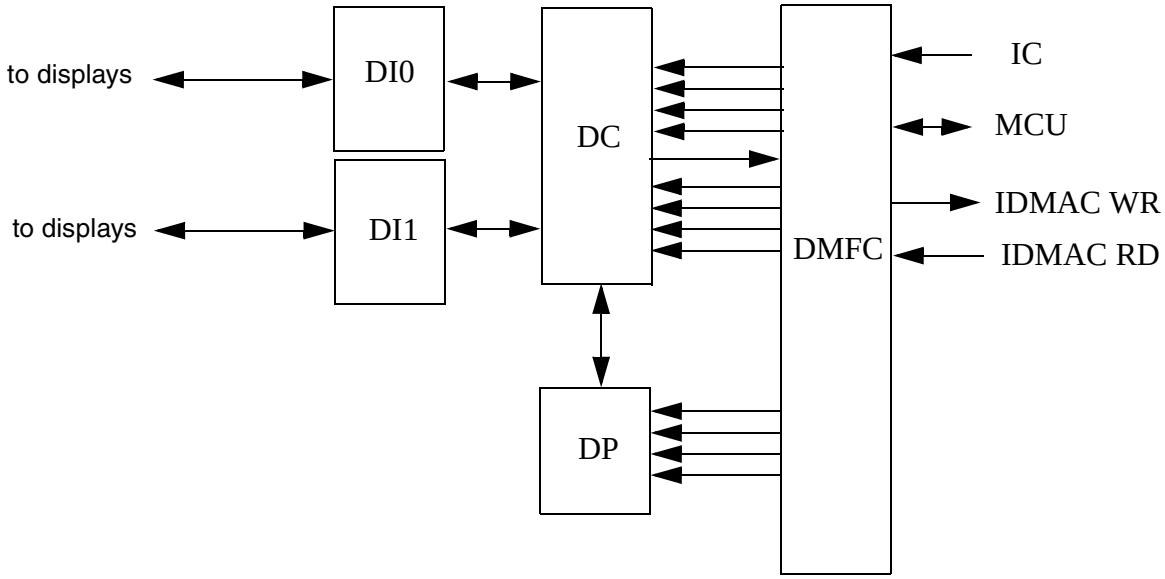


Figure 42-432. The display port

42.3.6.1 Display ports channels

The display port send data to the display over several channels. The data source may be the system's memory (via the IDMAC) the IC module and the MCU. The display port can read data from the display and send it to the MCU or IDMAC.

The data is routed over channels. The table below maps the IDMAC channels to DMFC, DC, DP channels and describes each channel.

Table 42-424. Display port channels

IDMAC's channel number	Display port's destination	DMFC channel number	DC channel number	Corresponding alpha channel	channel usage
21	DC	Programmable using dmfc_ic_in_port	Programmable using dmfc_ic_in_port	NA	This channel is coming from the IC module. When data is coming from the IC, the IC channel replaces one of the IDMAC's channels connected to the DMFC. When the IC_DMFC_SEL bit is set the output of the IC is routed to the DMFC. This channel can be routed to the DC channels 1,2,5B,5F,6B,6F. Routing this channel to the DC channel is done with the dmfc_ic_in_port bits. The DC's channel allocated to the IC channel can't be used for data coming from other source.
23	DP	5B	5	51	This channel is for the DP's primary flow. When a single plane is used, the data should be sent over this channel. When 2 planes are combined in this flow, the second plane should come on channel 27.
24	DP	6B	6	52	This channel is for the DP's secondary flow. When a single plane is used, the data should be sent over this channel. When 2 planes are combined in this flow, the second plane should come on channel 29.
27	DP	5F	5	31	This channel is for the DP's main flow. When a single plane is used, the data should be sent over channel 23 and this channel should not be used. When 2 planes are combined in this flow, one plane should come on channel 23 and the other plane on this channel.
28	DC	1	1	NA	This channel can serve sync and async flows. When channel 28 is connected to DI0, channel 23 must be connected to DI1 even if ch23 is not used. This is done by programming the PROG_DISP_ID_5 field
29	DP	6F	6	33	This channel is for the DP's secondary flow. When a single plane is used, the data should be sent over channel 24 and this channel should not be used. When 2 planes are combined in this flow, one plane should come on channel 24 and the other plane on this channel.
40	DC	0	0	NA	This is a read channel

Table 42-424. Display port channels

IDMAC's channel number	Display port's destination	DMFC channel number	DC channel number	Corresponding alpha channel	channel usage
41	DC	2	2	NA	This channel can serve only async flow
42	DC	1C		NA	Command stream. See 42.3.6.6, "Display port's restrictions
43	DC	2C		NA	Command stream. See 42.3.6.6, "Display port's restrictions
44	DC	3		NA	Mask channel. This mask channel can be associated with channel 23 or channel 28

42.3.6.2 Supported display interfaces

- The display port has 2 DI interfaces. Each interface can handle up to 3 displays.
- The total number of supported displays is 4.
- Each DI can handle up to 2 async interface - only one of them can be serial interface.
- Each DI can handle one synchronous interface. Asynchronous displays that are accessed in synchronous mode are considered synchronous interface.

42.3.6.2.1 Synchronous Interfaces

The DI supports the following synchronous display interfaces.

1. Synchronous generic interfaces to TFT dumb displays or RGB interfaces of smart displays.
2. Synchronous interfaces to Sharp displays.
3. Synchronous interfaces to TV encoders:
   a) PAL
   b) NTSC

   TV interfaces can operate in progressive or interlaced modes.
4. Synchronous interface to a graphic accelerator
5. BT.656
6. BT.1120

BT.1120 and BT.655 support

BT.1120 and BT.656 are supported. Only video data is supported, sending data during blanking intervals is not supported. The component size is always 8 bit.

42.3.6.2.2 Asynchronous Parallel Interfaces

The DI has a flexible asynchronous interface. The interface include 2 chip selects (CS) and 7 general purpose control signals. The user can decide which of the 7 signals will be associated to each one of the

asynchronous displays (up to 2 displays per DI). The using can configured some of the control signals to be shared by more than one display.

42.3.6.2.3 Asynchronous Serial Interfaces

See Section 42.3.10.5, “Serial display interface”

42.3.6.3 Display port’s bandwidth

When the IPUv3EX clock (HSP_CLK) is equal to 133Mhz, the peak bandwidth supported by the display port is as follows

- 110 Mega-Accesses/Sec if the DI clock is derived from an external to IPUv3EX source (like another PLL)
- 120 Mega-Accesses/Sec if the DI clock is derived from the IPUv3EX clock (HSP_CLK)

An access can be a pixel, component or generic data.

42.3.6.4 Display Dual Mode

This mode is useful for smart display with synchronous interface. The data is sent to the display only when the data has changed or when the display processor’s settings are changed. The physical interface to this display has synchronous interface’s characteristics.

42.3.6.5 Display Errors

There are few types of errors that may impact the image sent to the display. The IPUv3EX provides some automatic mechanisms allowing the system to overcome these errors.

42.3.6.5.1 Data starvation errors

These type of errors may happen for synchronous displays if the data is ready at the DI to be sent to the display at the time point it is required. This anomaly may happen if the system is very loaded and the IDMAC was not able to read data from the external memory and provide it to the DMFC.

Frame boundary errors

In case that a new frame should be sent to the display but the data of the previous frame was not sent completely, the IPUv3EX will reset the display modules and flush the internal buffers. As the IPUv3EX expects that the new frame indication will be triggered at least 2 lines (blanking interval) before the actual data is to be sent. The should recover and be ready with the data of the new frame within the blanking interval period.

Error within a frame

In case that a pixel was missed within a frame i.e. the pixel did not arrived to the DI on time. The IPUv3EX has 2 ways to handle the situation. The mode is selected by configuring the DI#_ERR_TREATMENT bit.

Redo the last access. The IPUv3EX will keep sending the last access to the display. The IPUv3EX will drop any pixel that arrive till the end of the line. If the data of the next line arrives correctly the IPUv3EX will continue working normally as overcame the problem. In case that the data of the next line is also incorrect the IPUv3EX will perform the same procedure as described on frame boundary errors.

Freeze the clock. In this mode the IPUv3EX freezes the clock sent to the display till the correct data arrives. In order to avoid a case where the clock is frozen forever and the system is stuck: when the clock is frozen, a watchdog timer starts counting. When the timer completed counting the IPUv3EX will perform the same procedure as described on frame boundary errors. The number of cycles to be counted by the watchdog timer is defined according to the DI#_WATCHDOG_MODE bits.

42.3.6.5.2 Anti tearing errors

In case of anti tearing errors, the IPUv3EX indicates about the error by asserting the corresponding DC_TEARING_ERR_# bit. See also Section 42.3.7.1.2, “Antitearing control”

42.3.6.6 Display port’s restrictions

- In case where 2 synchronous flows are used and additional asynchronous flow via DP is used. The asynchronous flow via DP can be targeted to the same DI that the synchronous flow via DP is targeted.
- There are only 2 command channels (42 and 42). Command channels are associated with data channels (24,28,41).
  - When channel 28 is associated with a command, the command stream will come from channel 42
  - When channel 41 is associated with a command, the command stream will come from channel 43
  - Channel 24 can be associated with a command stream only if channel 28 or channel 41 do not use a command stream. If channel 28 is not associated with a command stream then channel 24 can be associated with channel 42. If channel 41 is not associated with a command stream and channel 28 is associated with a command stream then channel 24 can be associated with channel 43.
- A channel that uses an alternate flow, cannot be associated with a command stream (ch. 24 or ch. 41)
- In case of a synchronous display using external clock, The DI where the synchronous display is connected to can be connected to another async display. But, it can support only the write direction. Read via the asynchronous interface cannot be performed if this DI uses external clock

42.3.7 DC - Display Controller

IPUv3EX handles few display flows supporting few displays. The IPUv3EX’s flows’ data sources can be the MCU, the system’s memory, a camera or an external device connected on the display’s port such as an external graphic accelerator. The data’s destination can be any device connected on one of the DI ports.

The DC controls the flows coming to and from the DI port. The DC manages the flows, decides which flows are currently active and when each flow is activated. The arbitrates between the active flows, gets the data from the predefined source and distribute it to the correct DI.

The DC's core is the microcode. The microcode contains a set of routine. A routine is built of a set of commands stored in the template's (microcode) memory. For each event (like new frame, end of frame etc.) a specific routine is executed. The user writes the routines and map them to a specific events. The routine contains instructions to the DC about the way of handling the data/address/commands associated with the display. The routine may contain information about the data's mapping, about waveform's characteristics, and more.

Figure 42-433 shows the micro architecture diagram for the DC block

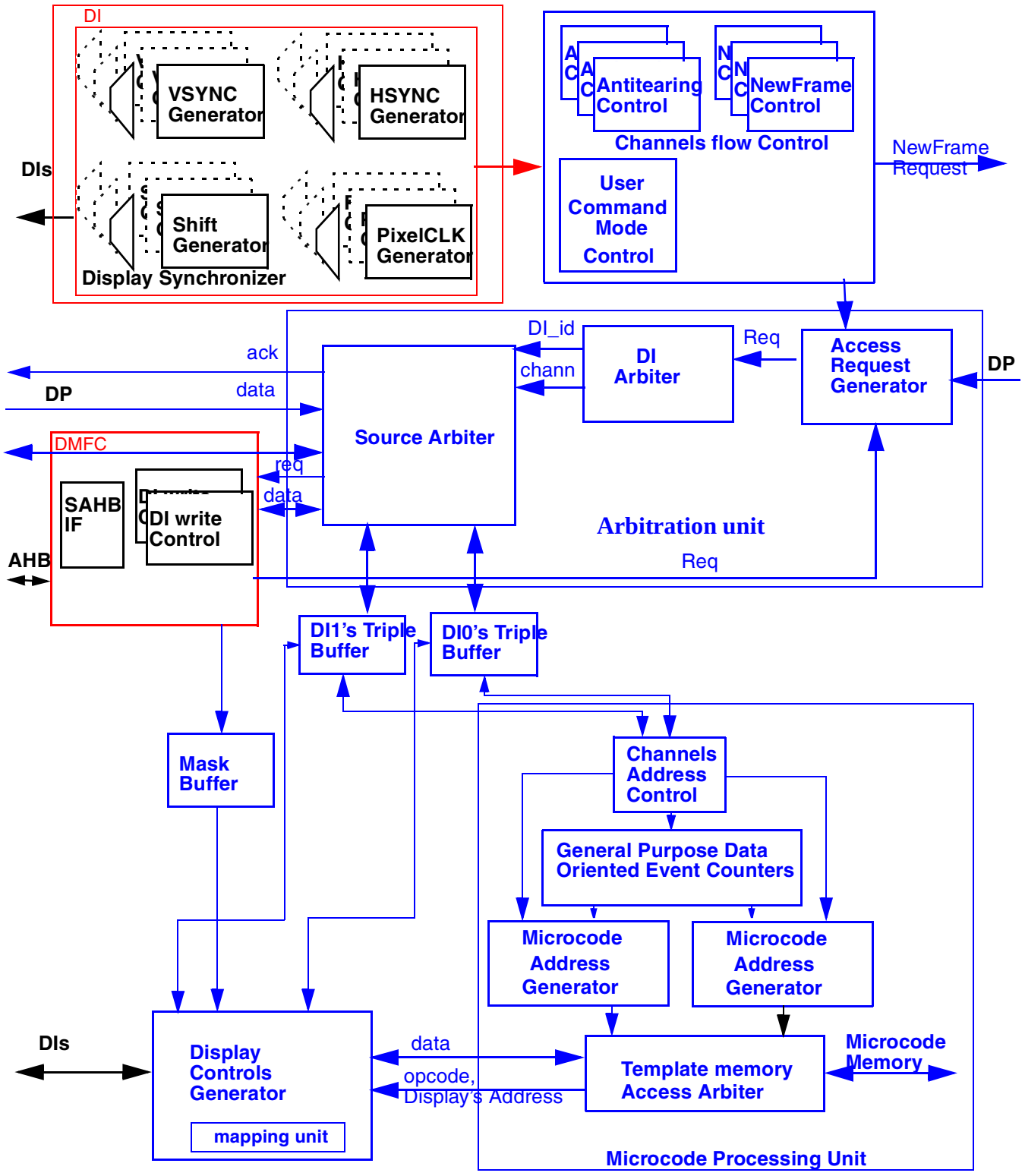


Figure 42-433. DC - Display Controller Block Diagram

42.3.7.1 Channels flow control

42.3.7.1.1 New Frame control

The channels flow control schedules the flows handled by the DC

For asynchronous flows the scheduling is done according to a request coming from the Frame Synchronization Unit (FSU) on the control module (CM).

For Synchronous flows the scheduling is done according to a trigger that is generated by a timer located on the one of the display's interface modules (DI)

42.3.7.1.2 Antitearing control

Anti tearing mechanism uses a signal indicating on a display's refresh of a frame. The supported tearing elimination triggers can be:

- An internally generated VSYNC signal
- A VSYNC signal generated by the display. The data source is the memory.
- A VSYNC signal coming from the CSI

The DC has the capability to avoid image tearing. For asynchronous flows where the source of the data is the system's memory or postprocessing, the DC monitors the position of a display's refresh pointer. Writing to the display is started only after crossing the window start point by the display refresh pointer. After that, writing to the display is allowed only when a write pointer does not advance beyond the refresh pointer. To provide anti-tearing mode, a window start time (in rows) must be defined in the PROG_START_TIME_1, PROG_START_TIME_2, PROG_START_TIME_5, PROG_START_TIME_6 registers for the corresponding channels.

The antitearing mechanism is limited to a case where only asynchronous flows are handle via the target DI.

In the case when tearing cannot be avoided (when the refresh rate is too high and the refresh pointer overtakes the write pointer after full refresh cycle), an error interrupt is generated. Tearing elimination mode can be disabled via the PROG_CHAN_TYP field for the corresponding channel.

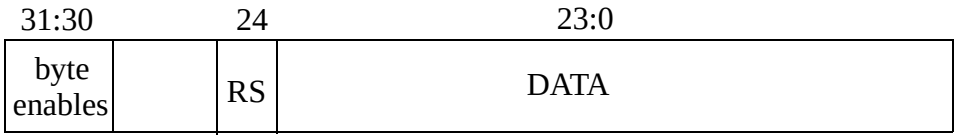
When the data arrives directly from the camera via IC, tearing can be avoided only for the 2:1 ratio between a sensor scan rate and a display refresh rate. The additional conditions for tearing elimination in this case are to synchronize the sensor and display VSYNCs (by programming the VSYNC's settings on the DI) and full-screen image from the camera via IC channel.

42.3.7.1.3 User command mode control

A user may prepare in the system's memory a buffer that holds commends to be sent to the external device. The command buffer includes the same amount of lines as the data buffer. The line of commands are sent to the display line by line. A line of data is sent following each line of commands. This mode is activated by programming the PROG_CHAN_TYP of the corresponding channel.

The structure of the command is as follows:

Figure 42-434. Structure of a command word



42.3.7.2 Arbitration Unit

This unit arbitrates the requests coming from the DMFC and from the DP and sends them to corresponding DI.

42.3.7.2.1 Access request generator

The requests coming from the DMFC or DP are sorted according to the target DI and then served according to a hard coded priority. The priority order is Sync flow, MCU access, IDMAC's Async flows.

42.3.7.2.2 DI arbiter

This units arbitrates between the DIs. The priority is hard coded. The priority order is Sync flow, MCU access, IDMAC's Async flows. In case of 2 simultaneous requests to different DIs with the same priority the requests are served in a Round-Robin fashion. In case of 2 simultaneous Sync flow requests to different DI, the user can bypass the Round Robin mechanism and prioritize one DI on the other according to the SYNC_PRIORITY_1 & SYNC_PRIORITY_5 bits. Setting low priority to both of these channels is forbidden.

42.3.7.2.3 Source arbiter

Once the source of the request to be served was selected and the target DI was chosen, this unit routes the request and all the signals, associated with it, to the correct triple buffer and to the correct source of the data.

42.3.7.3 Microcode processing unit

The main control unit of the DC is the Microcode processing unit (MPU). The data coming to the DC may be associated with some additional information like new frame, new line, new address etc. The information is processed in the MPU. The MPU executes the associated routine. The routine includes a set of instructions of the actions to be performed by the DC and DI.

42.3.7.3.1 Channels address control

This unit controls the display's address for each channel. This unit is responsible of defining the next address to be accessed (by incrementing or jumping). Stores special events flags(like EOF, EOL etc.). Based on the display's address and the special events the type of the routines to be executed is defined.

42.3.7.3.2 General purpose Data oriented events counters

A user may define up to 4 general purpose events. The events are triggered by a standard event (NF, NL). The standard event restarts a counter, when the counter completes counting the user's general purpose event is asserted. This event activates a routine like any other events.

42.3.7.3.3 Microcode address generator

This unit calculates the physical address of the template memory where the event associated routine resides. In addition these unit arbitrates between simultaneous events and select the event to be served. The arbitration is done according to a user defined priority. The priority of each event is set according to the #_CHAN_PRIORITY_CHAN_# bits of each channel. There are 2 modes of arbitration

- Serving all the pending events according to the priority
- Serving only the highest priority event while ignoring all the other events

The arbitration mode is defined according to the CHAN_MASK_DEFAULT bit of each channel.

42.3.7.3.4 Template's Memory Access Arbiter

This unit gets memory access requests from the 2 microcode address generator units and arbitrates between them in a Round Robin fashion. In case that only one of the requests belongs to a synchronous flow, this request is selected.

42.3.7.4 DC's Template structure

The template memory contains 256 template words. Each template word is a 42 bits words. Accessing a template word require 2 accesses (32bit each).

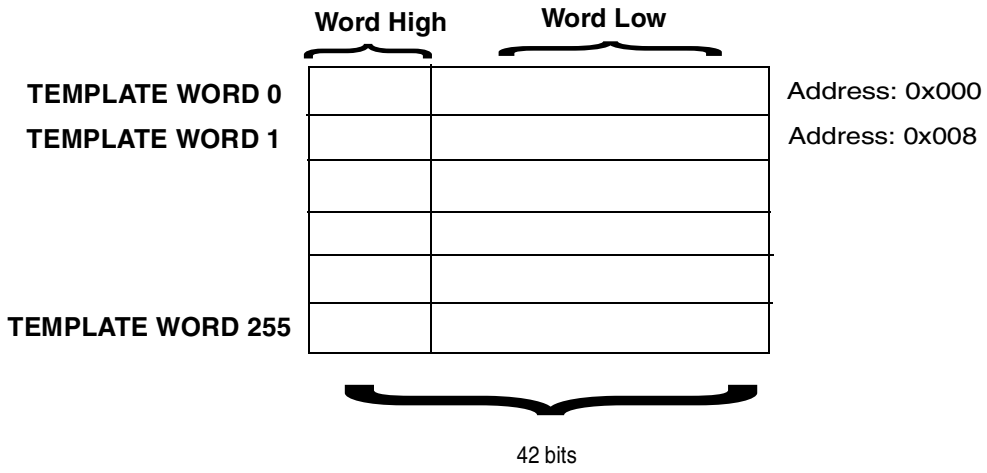


Figure 42-435. Template's structure

42.3.7.4.1 DC template's memory map

0x1F080000 DC_MICROCODE_W0_L																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	OPERAND												MAPPING			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MAP PING	WAVEFORM				GLUELOGIC							SYNC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1F080004 DC_MICROCODE_W0_H																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	STO P	OPCODE								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 42-436. DC template's memory map

DC template's fields description

Table 42-425. DC template's fields description

Field	Description
STOP	Stop bit - This bit should be set in order to indicate that the current command is the last command of the routine
OPCODE	The command's code
OPERAND	The command's operand - for some of the commands this field can hold a parameter associated with the command
MAPPING	The MAPPING field holds a pointer to a register holding 3 fields: MAPPING_PNTR_BYTE0_X, MAPPING_PNTR_BYTE1_X, MAPPING_PNTR_BYTE2_X. This pointers point to sets of OFFSET and MASK parameters that define the mapping scheme. MAPPING = 0 means that mapping is disabled. The value in this field should be incremented by 1 to get the correct X pointer value In order to point to MAPPING_PNTR_BYTE2_0, MAPPING_PNTR_BYTE1_0, MAPPING_PNTR_BYTE0_0 the user should write 1 to the MAPPING field

Table 42-425. DC template's fields description

Field	Description
WAVEFORM	For data oriented output pins. The IPUv3EX has 4 waveform generator units. The IPUv3EX holds 12 sets of waveforms' configuration registers called DI0_DW_GEN_<i> and DI1_DW_GEN_<i> The WAVEFORM field defines which one of the 12 waveforms' configuration registers is used. The DI1_DW_GEN_X register holds a pointer to one of the 4 waveform generators units for each of the data oriented pins. 0 - The waveform of the data oriented output pins is not affected 1 -Points to DI0_DW_GEN_0 or DI1_DW_GEN_0 2 - Points to DI0_DW_GEN_1 or DI1_DW_GEN_1 ... 12 - Points to DI0_DW_GEN_11 or DI1_DW_GEN_11

Table 42-425. DC template's fields description

Field	Description
GLUELOGIC	For signals generated by waveform generator #3; This field provides extra flexibility on the signals waveform
	GLUELOGIC[6] - This bit defines if we are in clock mode (1) or CS mode(0). 1- clock mode When the command is related to the display clock's pin then only if we are in clock mode, GLUELOGIC[5:4] are valid. 0- CS mode When the command is related to the CS pin then only if we are in CS mode, GLUELOGIC[3:0] are valid.
	GLUELOGIC[5:4] - clock mode settings 00 - Freeze the display clock following the execution of the current command 01 - Freeze the display clock before the execution of the next command to be executed 10 - Enable (unfreeze) the display clock following the execution of the current command 11 - Enable (unfreeze) the display clock before the execution of the next command to be executed
	GLUELOGIC[3] - CS mode settings 1 - Once the signal is asserted then it remains asserted (high or low according to the polarity) 0 - No impact on the waveform
	GLUELOGIC[2] - CS mode settings 1 - Once the signal is negated then it remains negated (high or low according to the polarity) 0 - No impact on the waveform
	GLUELOGIC[1] - CS mode settings 1- The current waveform can be attached to the previous waveform. If the previous waveform was asserted and the current waveform start asserted the two waveforms will be attached so the signals' waveforms will be consecutive. This impact the behavior of the previous waveform. This can be done only if GLUELOGIC[0] of the previous waveform is set to 1 0 - No impact on the waveform
	GLUELOGIC[0] - CS mode settings 1 - this bit allows the next waveform to be attached to the current waveform.
SYNC	The data associated with this command should be synchronized to the DI's one of gen_time_sync generators' output. 0000 - No sync. The data is sent without any synchronization to any event 0001 - Sync with unit #1 0010 - Sync with unit #2 ... 1010 - Sync with unit #10 1011 - 1111 Reserved

DC template's command description

The diagram below illustrates the command processing flow. The command is being fetched from the microcode memory. The Opcode is decoded. According to the decoding several controls are sent to the

processing unit. In addition the processing unit gets the pixel's data and the display's address. The processed data/ address is stored in an internal register. There are 2 types of commands

HOLD - In this type of commands the data/address is stored in the register and not sent to the DI. The data/address is held in the register for further processing in the following commands.

WRITE - In this type of commands the data/address is stored in the register and also sent to the DI.

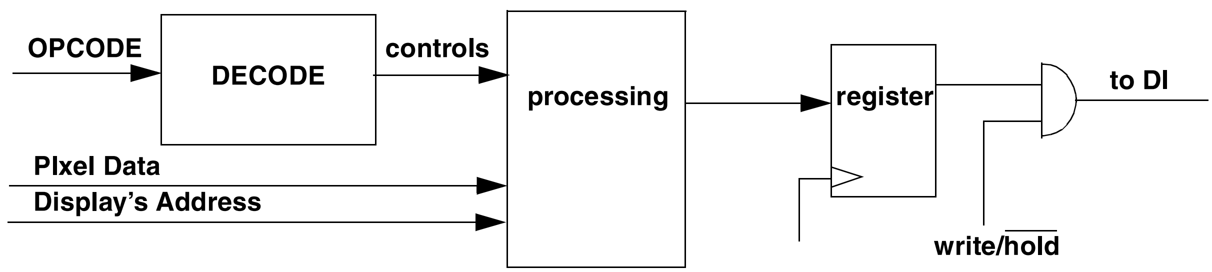


Figure 42-437. Opcode processing

The table below describes the DC's Commands.

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLG	S	0	0	0	0	DATA																																0	0	0	0	0
	Hold 32bit word in an internal register for further processing DATA is a general purpose data to be held																																									
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRG	S	0	1	DATA																								WAVEFORM				GLUELOGIC							SYNC			
	Write 24bit word to the DI and Hold the word in register DATA is a general purpose data to be written																																									
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLOA	S	1	0	1	0	af	DATA																MAPPPING					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Hold the display's address in an internal register for further processing af=0: No shift; af=1: 8 bit right shift. DATA is a 16bit general purpose data, This data is ORed with the 16 MSB of the mapped address																																									

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																														
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WROA	S	1	1	1	0	af	DATA																MAPPPING					WAVEFORM				GLUELOGIC								SYNC						
	Write address to the display and Hold address in register af=0: No shift; af=1: 8 bit right shift.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLOD	s	1	0	0	0	0	DATA																MAPPPING					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Hold data in register DATA is a 16bit general purpose data,. This data is ORed with the 16 MSB of the mapped data coming from the data's source IDMAC or MCU.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WROD	s	1	1	0	0	0	DATA																MAPPPING					WAVEFORM				GLUELOGIC								SYNC						
	Write data to DI and Hold data in register DATA is a 16bit general purpose data, This data is ORed with the 16 MSB of the mapped data coming from the data's source IDMAC or MCU.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLOAR	S	1	0	0	0	1	1	1	af	0	0	0	0	0	0	0	0	0	0	0	0	0	MAPPPING					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Adding Mapped Address to held data and hold in an internal register. af=0: No shift; af=1: 8 bit right shift.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WROAR	S	1	1	0	0	1	1	1	af	0	0	0	0	0	0	0	0	0	0	0	0	0	MAPPPING					WAVEFORM				GLUELOGIC								SYNC						
	Adding Mapped Address to held data, write to DI and hold in register af=0: No shift; af=1: 8 bit right shift.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HLODR	S	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MAPPPING					0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Adding Mapped Data to held data and hold in register																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																								
WRODR	S	1	1	0	0	1	1	0	0	M2	M1	M0	0	0	0	0	0	0	0	0	0	0	MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Adding Mapped Data to held data, write to DI and hold in register M0: 0: a new data[7:0] before mapping, 1: a previous access data [7:0] before mapping. M1: 0: a new data[15:8] before mapping 1: a previous access data [15:8] before mapping. M2: 0: a new data[31:16] before mapping 1: a previous access data [31:16] before mapping. if (M2 = M1 = M0 = 0) than the command performs: Adding a new Mapped Data to a held data in an internal register and write it to display else a display's data will be combined from a new data and previous data according to M-flags. The combined data will be mapped according to MAPPING and sent to a display. Examples: previous_data = 0x89ABCDEF new_data = 0x12345678 held_data (previous_data after mapping) = 0x0000EF Current MAPPING mode: new_data & 0x00ffff00 Output: if M2 = M1 = M0 = 0) than: Output = new_data OR held_data = 0x3456EF if M0 =0, M1=1,M2=1 than: Output = MAPPING ({previous_data[31:8],new_data[7:0]}) = MAPPING(0x89ABCD78) = 0x00ABCD00) if M0=0, M1=1,M2=0 than: Output = MAPPING ({new_data[31:16],previous_data[15:8],new_data[7:0]}) = MAPPING(0x1234CD78) = 0x0034CD00) if M0=1, M1=0, M2=0 than: Output = MAPPING ({new_data[31:8],previous_data[7:0]}) = MAPPING(0x123456EF) = 0x00345600)																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WRBC	S	1	1	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Merge 1 bit mask channel with mapped data Hold in register and write to DI Mask data is coming from the DC's mask channel.																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																								
WCLK	S	1	1	0	0	1	0	0	1	N_CLK_OPERAND													0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Wait N_CLK_OPERAND clocks N_CLK_OPERNAD is the number of DI_CLK cycles to wait.																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS-I	0	1	0	0	0	1	0	0	1	N_CLK_OPERAND													MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Wait for Status I Read from the display and compare to a predefined PASSWORD. If the read data is equal to the PASSWORD then continue if not redo the read & compare cycle. N_CLK_OPERAND is the number of DI_CLK cycles to wait before latching the data coming from the DI. DI_READ_DATA_ACK_VALUE_0 DI_READ_DATA_MASK_0																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS-II	0	1	0	0	0	1	0	1	0	N_CLK_OPERAND													MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Wait for Status II Read from the display and compare to a predefined PASSWORD. If the read data is equal to the PASSWORD then continue if not redo the read & compare cycle. N_CLK_OPERAND is the number of DI_CLK cycles to wait before latching the data coming from the DI. WSTS-II command must be followed by a WSTS-I command. This command is useful in case that the PASSWORD is read in 2 accesses. The comparison will be done only after the execution of the WSTS-I command.																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WSTS-III	S	1	0	0	0	1	0	1	1	N_CLK_OPERAND													MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Wait for Status III Read from the display and compare to a predefined PASSWORD. If the read data is equal to the PASSWORD then continue if not redo the read & compare cycle. N_CLK_OPERAND is the number of DI_CLK cycles to wait before latching the data coming from the DI. WSTS-III command must be followed by a WSTS-II command. This command is useful in case that the PASSWORD is read in 3 accesses. The comparison will be done only after the execution of the WSTS-I command. In case of a PASSWORD mismatch the read is done again. The entire cycle (WSTS-III -> WSTS-II -> WSTS-I) will be performed.																																								
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD	S	1	0	0	0	1	0	0	0	N_CLK_OPERAND													MAPPPING					WAVEFORM				GLUELOGIC							SYNC			
	Read data from DI N_CLK_OPERAND - means delay value in DI_CLK for display's data latching by DI, defined by user For serial display the N_CLK_OPERAND is fixed and should be set to 1 value.																																								

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																														
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WACK	S	1	0	0	0	1	1	0	1	0	N_CLK_OPERAND													0	0	0	0	0	WAVEFORM				GLUELOGIC								SYNC			
	Wait for acknowledge N_CLK_OPERAND - Number of DI_CLK cycles to count before monitoring the ACK received from the display.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSK	S	1	1	0	0	1	0	0	0	0	0	0	0	0	e0m	e1m	e2m	e3m	nfm	nlm	nfldm	eofm	eolm	eofldm	nadm	ncm	dm	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Mask - Mask a specific event; In case of more than one pending events the user can mask some of the event and serve the others according to the priority. e0m - event 0 mask, defined by user e1m - event 1 mask, defined by user e2m - event 2 mask, defined by user e3m - event 3 mask, defined by user nfm - new frame mask, defined by user nlm - new line mask, defined by user nfldm - new field mask, defined by user eofm - end of frame mask, defined by user eolm - end of line mask, defined by user eofldm - end of field mask, defined by user nadm - new address mask, defined by user ncm - new channel mask, defined by user dm - data mask, defined by user																																														
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HMA	S	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ADDRESS				0	0	0	0	0							
	HOLD_MICROCODE_ADDRESS - hold operand in special Microcode address register 0. This register is used to store a pointer in the microcode to be used by the BMA command ADDRESS is a microcode address in the microcode memory defined by the user.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HMA1	S	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ADDRESS				0	0	0	0	0							
	HOLD_MICROCODE_ADDRESS - hold operand in special Microcode address register 1. This register is used to store a pointer in the microcode to be used by the BMA command ADDRESS is a microcode address in the microcode memory defined by the user.																																													
	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table 42-426. DC template's commands description

Command	COMMAND[41:0]																																
BMA	S	0	0	1	1	LF	AF	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	N		0	0	SYNC	
	BRANCH_MICROCODE_ADDRESS jump to Microcode address which is stored at an internal Microcode address register. The branch will be done to the ADDRESS stored with the HMA/HMA1 command. The BMA command manages the repeat cycles by one of the auto decrement counters (internal counters), the counter's id is defined by LF-bit (0/1) The N operand defines the N-number of routine's repetitions. In case of N=0. The BMA command functions as a JUMP command. As long as the internal counter didn't reached its predefined value, the microcode will branch to the pre defined address stored by the HMA/HMA1 commands. Selecting between HMA and HMA1 is done by the AF bit (it is calculated according to the "1 - AF" value). - That's the subroutine's microcode start address When the counter reached its predefined value, the microcode will branch to the pre defined address stored by the HMA/HMA1 commands. Selecting between HMA and HMA1 is done by the AF bit (it is calculated according to the "AF" value). - That's the subroutine's microcode return address BMA synchronization (SYNC field): The BMA command can be used for context switching between 2-channels. When switching between flows the user can run a routine for a new channel, which actually affects the previous flow. This feature is useful when a new flow breaks the current flow (post command for example): SYNC is used for display_id selection // 0 0 0 - current display // 0 0 1 - previous display // 1 0 0 - display 0 // 1 0 1 - display 1 // 1 1 0 - display 2 // 1 1 1 - display 3																																

42.3.7.5 Display controls' generator

This unit generates the control signals associated with the data that are sent to the DI module.

42.3.7.5.1 Bus Mapping Unit

The Bus Mapping Unit is responsible for programmable mapping of the input data and commands to the display interface format and vice versa. Address mapping is done by this unit as well (programmable via micro code). The internal DI format for data and commands is a 24-bits word divided into three byte components (eight zeroes are added to MSB for 16-bits words from the DC). This word can be output or input in one, two, three or four cycles of the display clock.

The word coming from the memory is a 32bit word. The mapping operation is done on 24 bits only. The 24 bit are selected from the received 32 bit according to the W_SIZE_# field. The 24 bit input word is partitioned to a 3 bytes (3X8bits). Each byte can be mapped to any position at the 24bit output word. The exact position is set according to the MD_MASK & MD_OFFSET fields.

The MAPPING_PNTR_BYTE0_X, MAPPING_PNTR_BYTE1_X, MAPPING_PNTR_BYTE2_X fields holds the pointers for the MD_MASK & MD_OFFSET for each byte.

The MAPPING field holds a pointer to a register holding 3 fields: MAPPING_PNTR_BYTE0_X, MAPPING_PNTR_BYTE1_X, MAPPING_PNTR_BYTE2_X.

0 - no mapping i.e. 32 bits are sent as is.

1 - points to MAPPING_PNTR_BYTE0_0, MAPPING_PNTR_BYTE1_0, MAPPING_PNTR_BYTE2_0

2 - points to MAPPING_PNTR_BYTE0_1, MAPPING_PNTR_BYTE1_1, MAPPING_PNTR_BYTE2_1

...

30 - points to MAPPING_PNTR_BYTE0_29, MAPPING_PNTR_BYTE1_29, MAPPING_PNTR_BYTE2_29

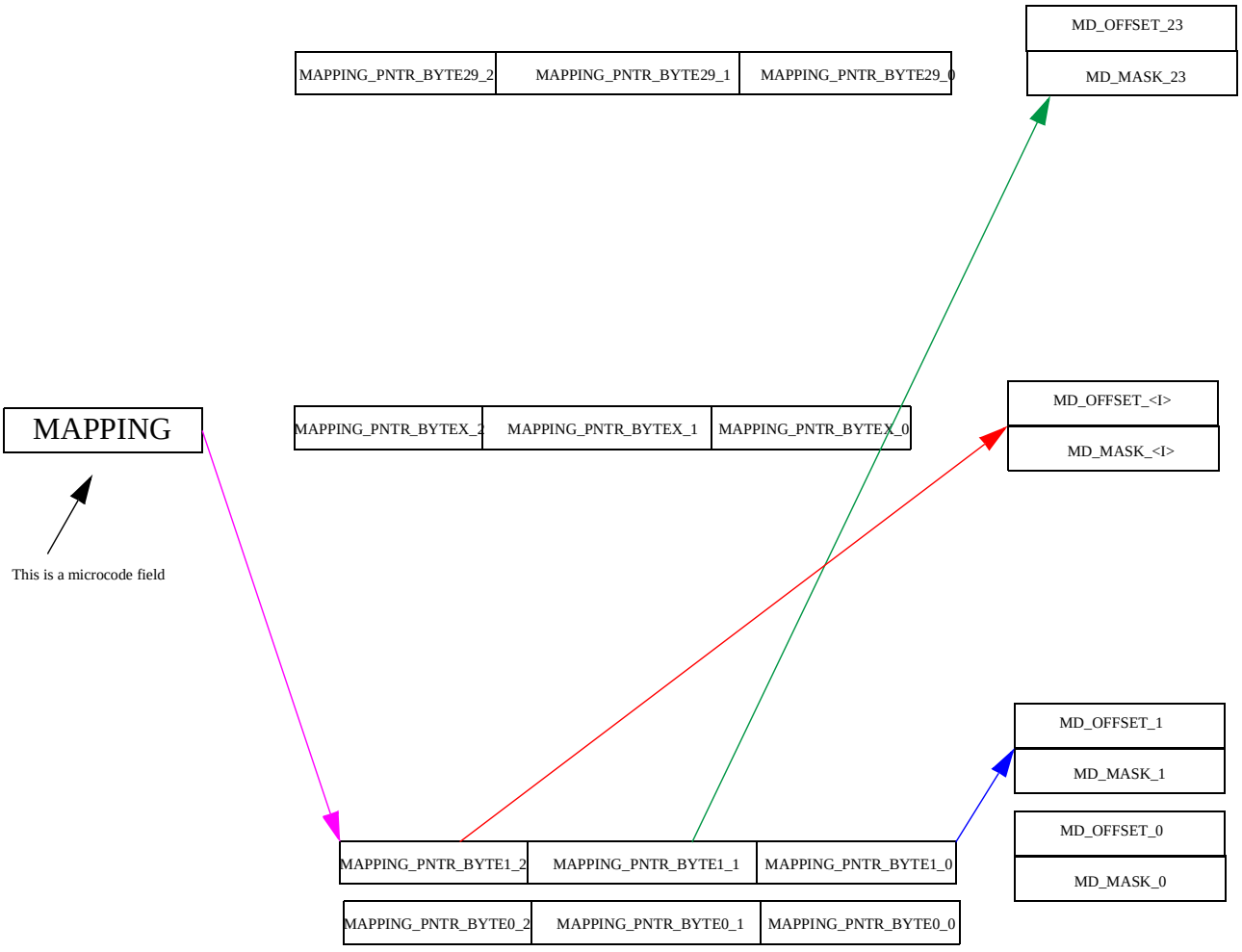
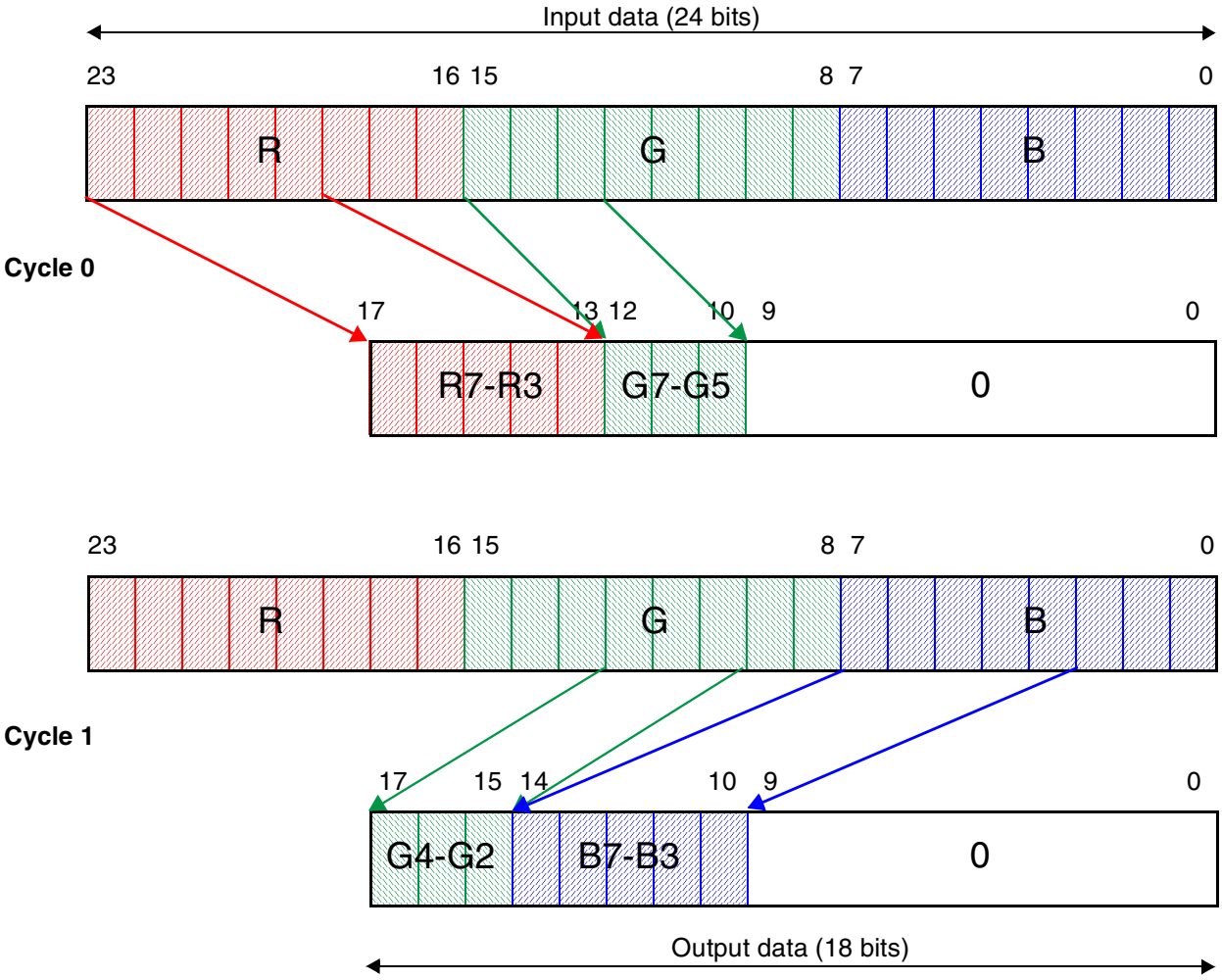


Figure 42-438. Mapping scheme

The mapping rule written in each of the Registers defines two types of parameters for the specific byte component and display:

1. Offsets of the byte component MSB relative to the output word LSB. Because the offsets can change dynamically, they are defined separately for the display clock cycles zero, one and two.
2. Numbers of the display clock cycles at which every bit of the byte component should be valid on the display bus. There are eight such 2-bit numbers in the Register.

Figure 42-439 presents an example of programming data packing for one of displays. Cycle 0 and Cycle1 in the diagram represent two separate atomic operations performed by the microcode template.



Cycle 0

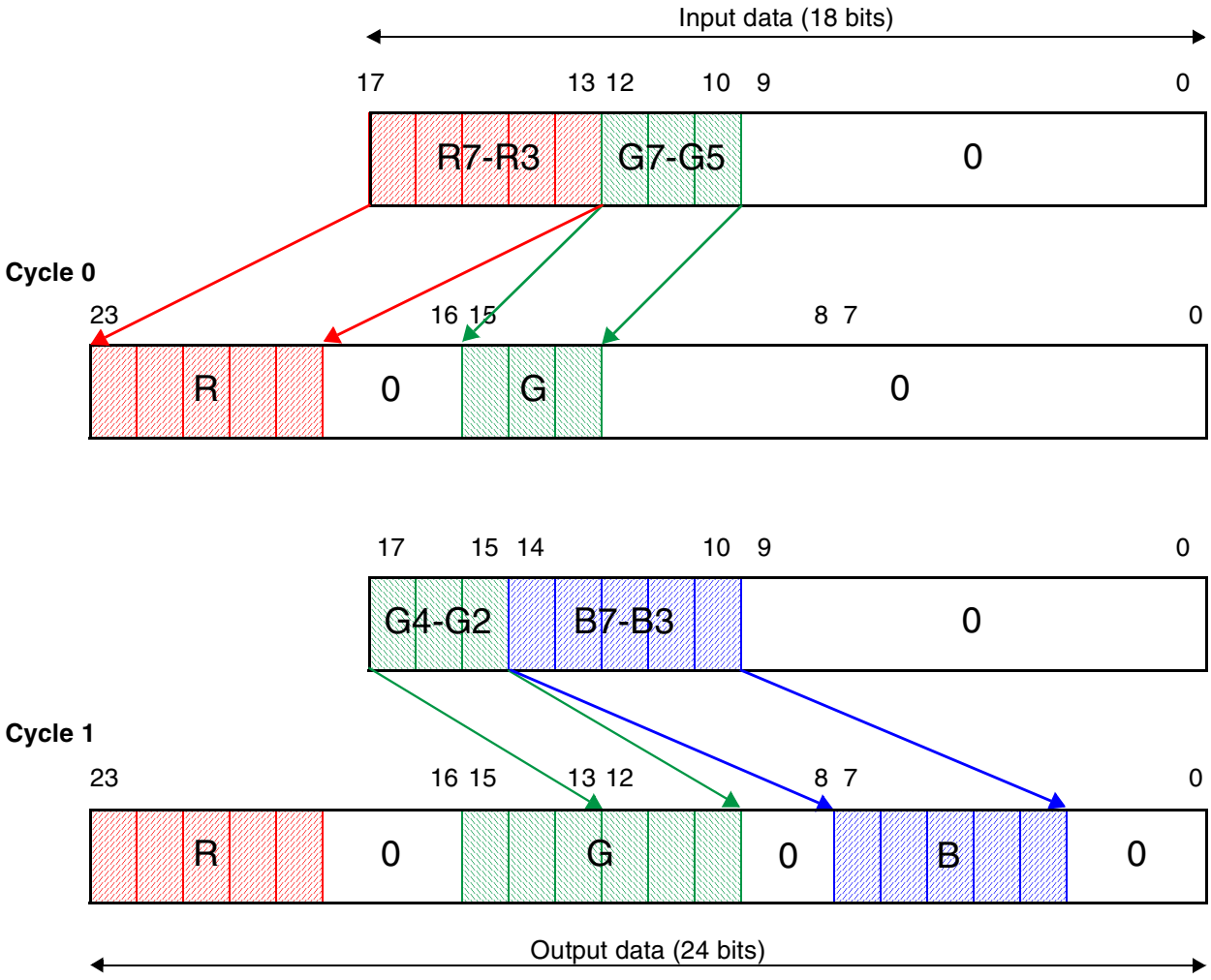
MD_OFFSET_[R] = 0x11 ; MD_MASK_[R] = 0xF8
MD_OFFSET_[G] = 0xC ; MD_MASK_[G] = 0xE0
MD_OFFSET_[B] = 0x0 ; MD_MASK_[B] = 0x0

Cycle 1

MD_OFFSET_[R] = 0x0; MD_MASK_[R] = 0x0
MD_OFFSET_[G] = 0x14 ; MD_MASK_[G] = 0x1C
MD_OFFSET_[B] = 0xE ; MD_MASK_[B] = 0xF8

Figure 42-439. Example of Data Packing for Writing Data to the Display

Figure 42-440 presents an example of programming data packing for one of displays. Cycle 0 and Cycle1 in the diagram represent two separate atomic operations performed by the microcode template.



Cycle 0

MD_OFFSET_[R] = 0x11 ; MD_MASK_[R] = 0xF8
MD_OFFSET_[G] = 0xC ; MD_MASK_[G] = 0xE0
MD_OFFSET_[B] = 0x0 ; MD_MASK_[B] = 0x0

Cycle 1

MD_OFFSET_[R] = 0x0; MD_MASK_[R] = 0x0
MD_OFFSET_[G] = 0x14 ; MD_MASK_[G] = 0x1C
MD_OFFSET_[B] = 0xE ; MD_MASK_[B] = 0xF8

Figure 42-440. Example of Data Unpacking for Reading Data from the Display

The same packing/unpacking registers are used for parallel and serial interface.

42.3.8 DMFC - Display Multi FIFO Controller

Figure 441 shows the block diagram for the DMFC block

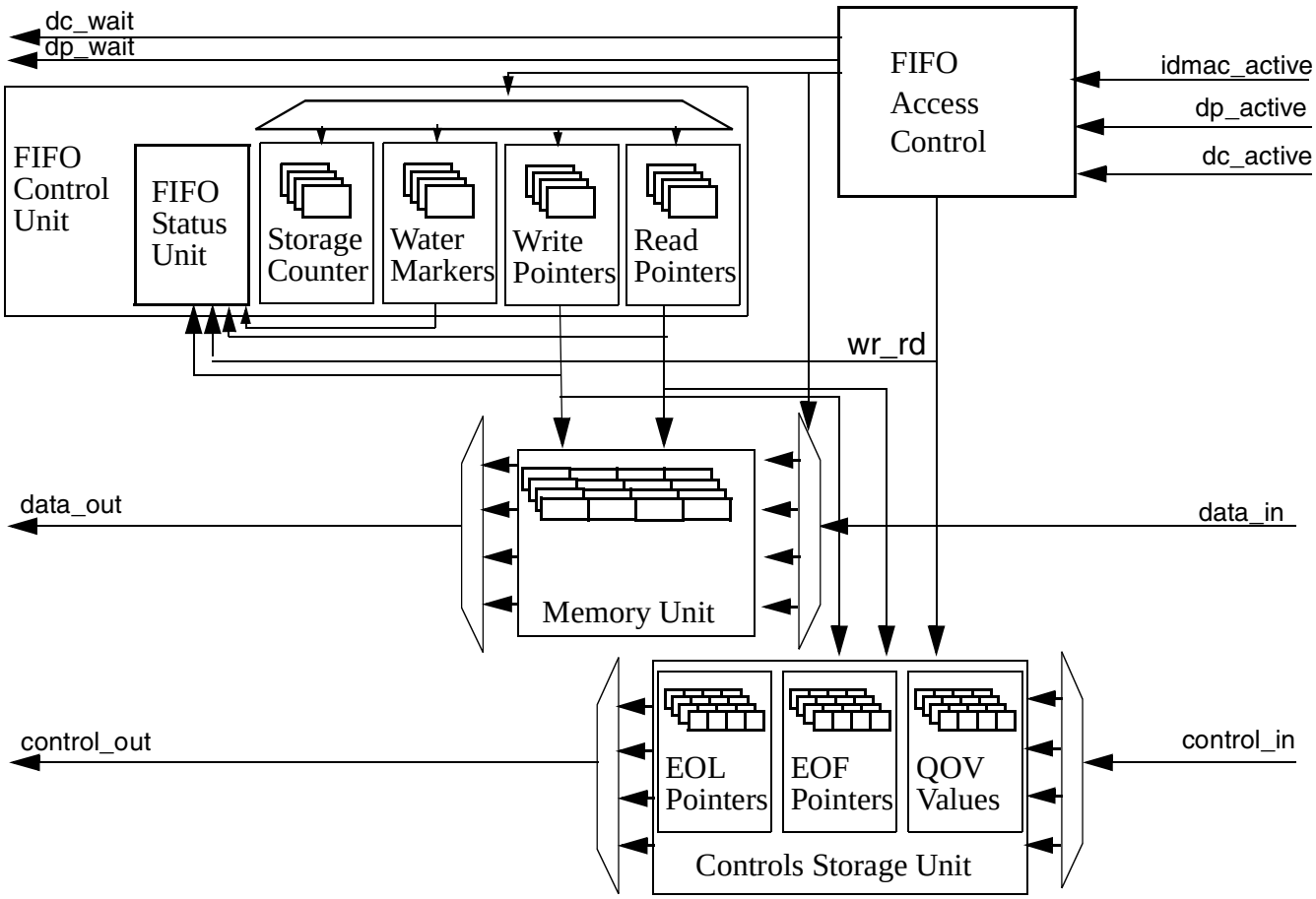


Figure 441. Display Multi Fifo Controller Block Diagram

The Display Multi Fifo Control manages Multi channels FIFOs. The DMFC serves the following clients

- IDMAC - both read and write
- DP - read only
- DC - both read and write
- IC - write only
- AHB - both read and write

The DP and the DC read channels are physically attached to an IDMAC or an IC channel. As the IC has only one output channel connected to the DMFC. When the input is coming from the IC it replaces a channel that was physically attached to the IDMAC. The DMFC uses a single physical memory that serves the DP and DC read channels. The AHB accesses to the DC and the DC's write channel (read from display) use a separate physical memory.

42.3.8.1 DP and DC read channels

Each one of the DP and the DC read (read from memory) channels is physically attached to an IDMAC read channel. A portion of the physical memory is allocated for each channel. The DMFC arbitrates between channels according to a predefined priority.

The DMFC controls each of the FIFOs

- Assert a request any time there's available place on the FIFO.
- Make sure that there's available place on the FIFO to accept the coming data.
- Optionally assert a watermark indication to avoid starvation

42.3.8.1.1 FIFO allocation to channels

The physical memory is partitioned to 8 segments. For each channel the user has to define the start address at a segment's boundary using the DMFC_ST_ADDR parameter. The size of the FIFO allocated to a channel is defined by the DMFC_FIFO_SIZE parameter. The user must allocate the FIFO and avoid overlapping between FIFOs.

The DMFC hold few special indications like EOF, EOL, EOFILD. The most important one is end of line (EOL). If the size of the FIFO is shorter than or equal to the IDMAC line's length (FW) than no special restrictions on the DMFC usage.

If the size of the FIFO is greater than the IDMAC line's length (FW) than the user has to be aware to the following restriction for each channel.

The DMFC has two operation modes which are distinguished by the wait4eot bit. For each channel the DMFC can store a maximum number of EOL indication. The maximum number of EOL indications of EOL is given on Table 42-427.

Table 42-427. DMFC's number of EOL indications

IDMAC's Channel	Maximum lines on the FIFO
23	Up to 3 lines
27	Up to 2 lines
28	Up to 2 lines
Other	Up to 1 line

If the use case is that the number of EOL indications cannot exceed the maximum number of EOL indications than the user should have the wait4eot cleared.

If the use case is that the number of EOL indication can exceed the maximum number of EOL indications than the user should have the wait4eot set.

Having the wait4eot bit set has performance impact as the DMFC analyzes the data prior to sending it to the destination. In addition the DMFC cannot utilize the entire FIFO allocated to this channel.

The user need to specify the burst size of the IDMAC by setting the DMFC_BURST_SIZE field. This field must match the IDMAC settings. In case that the IDMAC's burst size is not a power-of-2 number, the value of this field should be rounded up to the nearest power-of-2 number. The burst size must not be greater than the FIFO's size.

42.3.8.1.2 Arbitration between channels

The arbitration between channels is fully hardware controlled. IDMAC has the highest priority. Then the synchronous channels. Then the asynchronous channels.

42.3.8.1.3 Watermark

The DMFC can generate a water mark signal for each channel. The watermark signal is sent to the IDMAC and dynamically increases the channels priority on the IDMAC's arbitration.

The watermark feature is enabled by the DMFC_WM_EN bit. The FIFO is partitioned to bursts. The user can set the watermark level at a burst boundary.

The watermark signal is set when the number of bursts on the FIFO + the number of already requested burst is smaller than the value specified on DMFC_WM_SET bit.

The watermark signal is cleared when the number of bursts on the FIFO + the number of already requested burst is greater than the value specified on DMFC_WM_CLR bit.

DMFC_WM_SET must be smaller than DMFC_WM_CLR.

42.3.8.2 IC interface

One of the IDMAC channels can be replaced by a flow coming from the IC using the DMFC_IC_IN_PORT. The user has to provide the IC's setting to the DMFC by programming the DMFC_IC_FRAME_WIDTH_RD, DMFC_IC_FRAME_HEIGHT_RD and DMFC_IC_FRAME_PPW_C fields. The burst size of the channel coming from the IC should always be programmed to 4 words.

42.3.8.3 DC write channel and AHB accesses

The second physical memory of the DMFC serves the

- IDMAC write channel (read from display)
- 2 AHB channels that can be read or write

The IDMAC write channel is programmed using the DMFC_RD_CHAN register in a similar way to the DC and DP read channels described above. The user has to provide the frame width and height for this channel and the pixel per word parameter.

The AHB channels are used from accesses via the AHB port to the display. The accesses are distributed between channels according to the MCU_T parameter.

42.3.9 DP - Display Processor

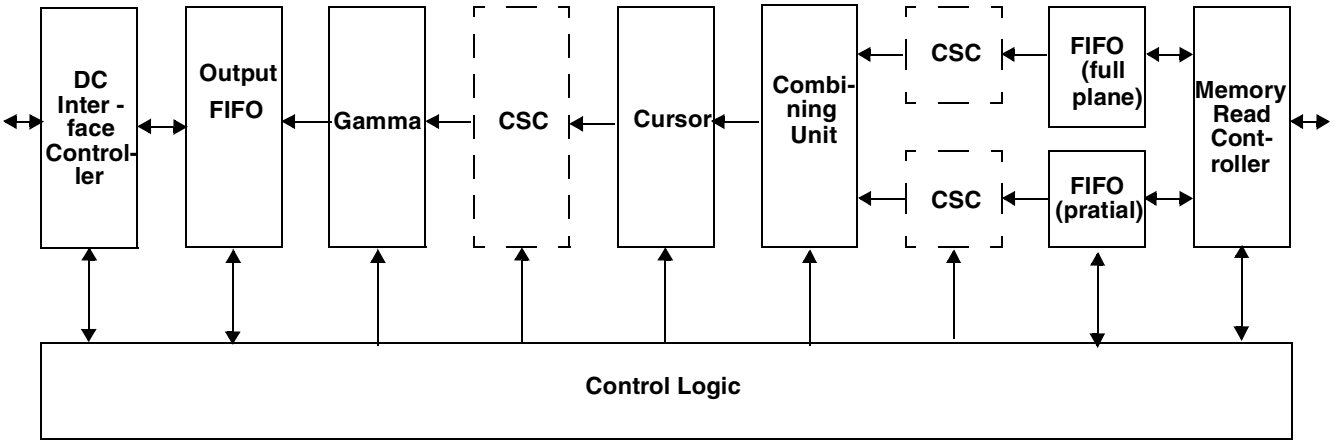


Figure 42-442. DP Micro architecture diagram

The display processor processes the image prior to sending it to the display. The main task performed by the DP is combining between 2 planes. The DP has 2 input FIFOs holding the data of full plane and the partial plane. In addition the DP performs some image enhancement functions like gamma correction, Color space conversion including Gamut mapping.

42.3.9.1 The DP programming model

The DP supports 3 flows. One sync flow and 2 Async flows. The DP holds 3 sets of registers. one set for each flow. Hence when referring to a register in this section the information is applicable to all the 3 sets. for example when referring to DP_COM_CONF, the information is applicable to DP_COM_CONF_SYNC, DP_COM_CONF_ASYNC0 and DP_COM_CONF_ASYNC1.

42.3.9.2 Displayed Planes

Figure 42-443 shows the planes displayed on a display. There are full and partial planes. The partial plane's position is defined relatively to the upper left corner of the full plane (BGXP and BGYP). The size of the partial and full planes is defined on the corresponding IDMAC's channels' FW and FH parameters. The cursor position and parameters are set in the DP_CUR_POS register.

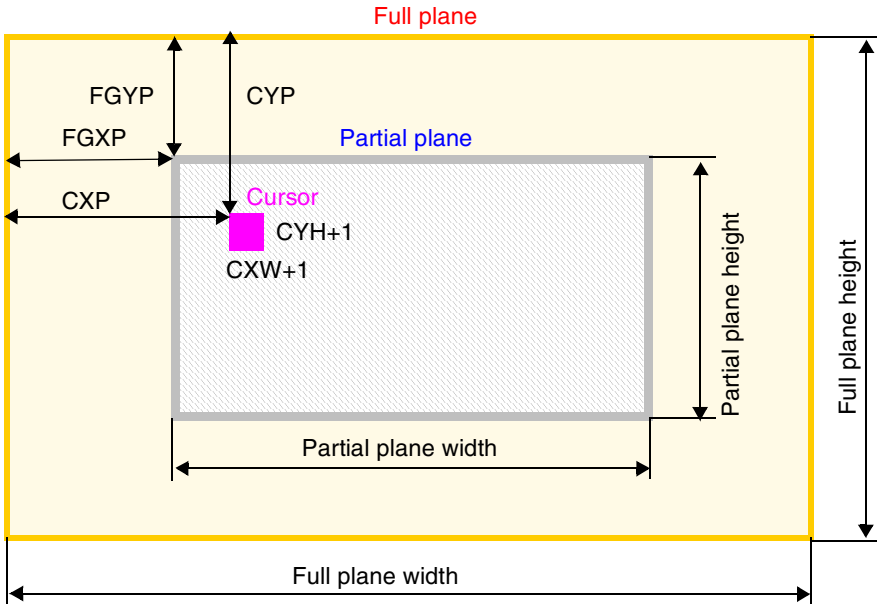


Figure 42-443. Displayed Planes

42.3.9.3 Combining Unit

The Combining Unit performs combining between the full and the partial planes. Each one of the planes may be graphics or video plane.

There are the following combining options:

- local alpha blending,
- global alpha blending,
- use of key color.
- order of the planes (full is presented over the partial plane and vice versa)

Combining mode is selected via the DP_COM_CONF Register. The combining equation is:

$$OP = BG*(1 - a) + FG*a \qquad \textit{Eqn. 42-4}$$

Where BG and FG are 2 input pixels; The DP_GWSEL bit defines if the BG is the pixel coming from the full plane or the partial plane.

a= (A+floor(A/128))/256 - an alpha value

A - a global or local transparency parameter.

The global A is written in the DP_GWAV field, the local A arrives together with the pixel.

A pixel becomes transparent when color keying is enabled and a pixel color matches a key color (independently on an alpha parameter). The color keying is defined on:DP_GWCKR, DP_GWCKG, DP_GWCKB

Combining takes 1 cycle per pixel. The Combining Unit outputs 24-bit words in the RGB/YUV format.

42.3.9.4 Cursor Generator

The Combining Unit output is passes through the Cursor Generator. The cursor's size and position are set via the DP_CUR_POS Register, a cursor color - via the DP_CUR_MAP Register. Different logic functions of combining the cursor with the image are supported as defined by the DP_COC field. The cursor can be blinking. The blinking mechanism resides on the display controller module. The blinking parameters are defined on the DC_BK_EN and DC_BKDVIV fields.

42.3.9.5 Color Space Conversion unit - CSC

The DP can get 2 input pixels from 2 different color spaces (YUV or RGB) and convert one of them to a common color space (YUV or RGB). In addition the 2 inputs can be of the same color space where the result is converted to another color space (YUV or RGB). The DP has a single CSC unit that can be placed on one of 3 locations:

- At the output of one of the 2 input FIFOs
- At the output of the cursor generator.

Placing is done according to the DP_CSC_DEF field.

The color conversion implements a 3x3 matrix multiplication between the full RGB pixels and the color conversion constants, in order to obtain a YCC format image.

The conversion formula is:

$$x \rightarrow Clip(Round(S * 2^{E})),\ S = Ax + B \qquad \textit{Eqn. 42-5}$$

where

A is a 3x3-dimensional matrix of weights, each a 10-bit signed number with 8 fractional digits

$$A = \begin{bmatrix} CSC\_A0 & CSC\_A1 & CSC\_A2 \\ CSC\_A3 & CSC\_A4 & CSC\_A5 \\ CSC\_A6 & CSC\_A7 & CSC\_A8 \end{bmatrix}$$

B is a 3-dimensional vector of offsets, each a 14-bit signed number with 2 fractional digits

$$B = \begin{bmatrix} CSC\_B0 & CSC\_B1 & CSC\_B2 \end{bmatrix}$$

S is a 3 dimensional vector of sums, each a 16-bit signed number with 4 fractional digits

$$S=Ax+B \quad \textit{Eqn. 42-6}$$

E is an exponent, assuming one of the following values: -1,0,1,2 (allowing weights up to 8). The CSC_S parameters are encoded by 2 bits, please refer to the CSC_S parameter description.

$$E = [CSC\_S0\ CSC\_S1\ CSC\_S2]$$

A more explicit formula:

$$S[i] = (sum(A[i][j]*In[j]) >> 4) + (B[i] << 2) + (1 << (3-E[i])) \quad \textit{Eqn. 42-7}$$

$$Out[i] = Clip(S[i] >> 4-E[i]) \quad \textit{Eqn. 42-8}$$

Where Clip() performs clipping to the range 0..255 (either per-component clipping or more sophisticated clipping that preserves the hue of the pixel)

42.3.9.5.1 Gamut mapping

When the color transformation produces colors outside the allowed range, they must be mapped back. This is called gamut mapping. The DP supports 2 clipping algorithms. (controlled by GAMUT_SAT_EN bit)

Hue preserving clipping algorithm is suitable only for RGB components. For YUV components, the per-component clipping algorithm is used.

Per component Clipping

This mapping is performed by clipping each of the components independently to its allowed range of values (the final value being uint8):

- Y: to 0..255 or 16..235 according to the SAT_MODE bit
- U/V: to 0.255 or 16..240 according to the SAT_MODE bit

Hue Preserving Clipping

Hue Preserving clipping is done in the following way

- First stage - eliminating negative values

```
N = min(R,G,B)
if (N<0) X-> X-N, where X=R,G,B
```

- At this stage, all components are non-negative and the MSB's beyond d=11 bits are ignored (assumed 0).
- Second stage - eliminating large values

```
M = max(R,G,B) (d-bit integer)
if (M>255)
M’ = M >> (m-7), where m=8..d-1 is the index of the most-significant non-zero
bit in M
D’ = Ceil(256*255/M’) = 256..510 (since M’ = 128..255)
Ceil(x) = the smallest integer which is not smaller than x
Implemented by a hard-wired 128x9-bit LUT
X -> min(255, (X*D’)>>(m+1)), where X=R,G,B (8x9 multiplier)
else
X -> X
```

42.3.9.6 Gamma correction

The DP includes a gamma correction function.

It is approximated bt the piece-wise polynomial:

$$\text{gammar} = (\text{GAMMA\_C<i>} + (\text{R[4:0]} * \text{GAMMA\_S<i>}) >> 7 \qquad \textit{Eqn. 42-9}$$

Where R is the input red componnent, that’s a 9 bit input composed pixel_in*2+pixel_in[7].

pixel_in is the 8 bit input pixel.

i is equal to MSB of R.

In order to calculate GAMMA_S<i> a following equation is used:

$$\text{GAMMA\_S[i]} = (\text{GAMMA\_C<i+1>} - \text{GAMMA\_C<i>})/16 \qquad \textit{Eqn. 42-10}$$

Single approximation slope is used for Gamma Correction of red, green and blue color components. However the Gamma Correction module instantiated three times since processing for color components should be done in parallel. The gamma transform is also available for changing the contrast of the luminance component only.

42.3.9.7 DC interface

The DC interface unit perform 2 tasks

- Starts the flow via the DP by getting a request from the DC and once the DP is ready send the new frame request to the IDMAC.
- Control the DP’s output FIFO and send the data to the DC using an handshake mechanism.

42.3.9.8 DP’s flows management

The DP can manage up to 3 flows: one synchronous flow and 2 asynchronous flows. However the DP can handle only one flow simultaneously. The DP has an automatic mechanism to control and switch between flows. The DP’s highest priority flow is the sync flow. An async flow can be executed during the blanking interval of the sync flow. An async flow can be stopped in order to serve the sync flow. The sync flow cannot be broken.The DP holds 3 sets of registers, one set for each flow. The registers are located in the

SRM memory. According to the flow that needs to be executed the correct set of registers is loaded to the DP.

Figure 42-444 illustrates the flow management done by the DP.

A flow starts when a request from the DC arrives and the internal pipe is empty. For sync flows the full frame NF (new frame) indication is immediately. In case that a partial frame is also used the NF indication will be sent one row before the row that the partial plane is actually positioned.

In case of ASYNC flow the DP first check if the previous async flow was broken. If yes, the DP restores the last settings of the previous flow. If this is a new async flow the DP will first reload the flow's parameters from the SRM. In case that a partial frame is also used the NF indication will be sent one row before the row that the partial plane is actually positioned.

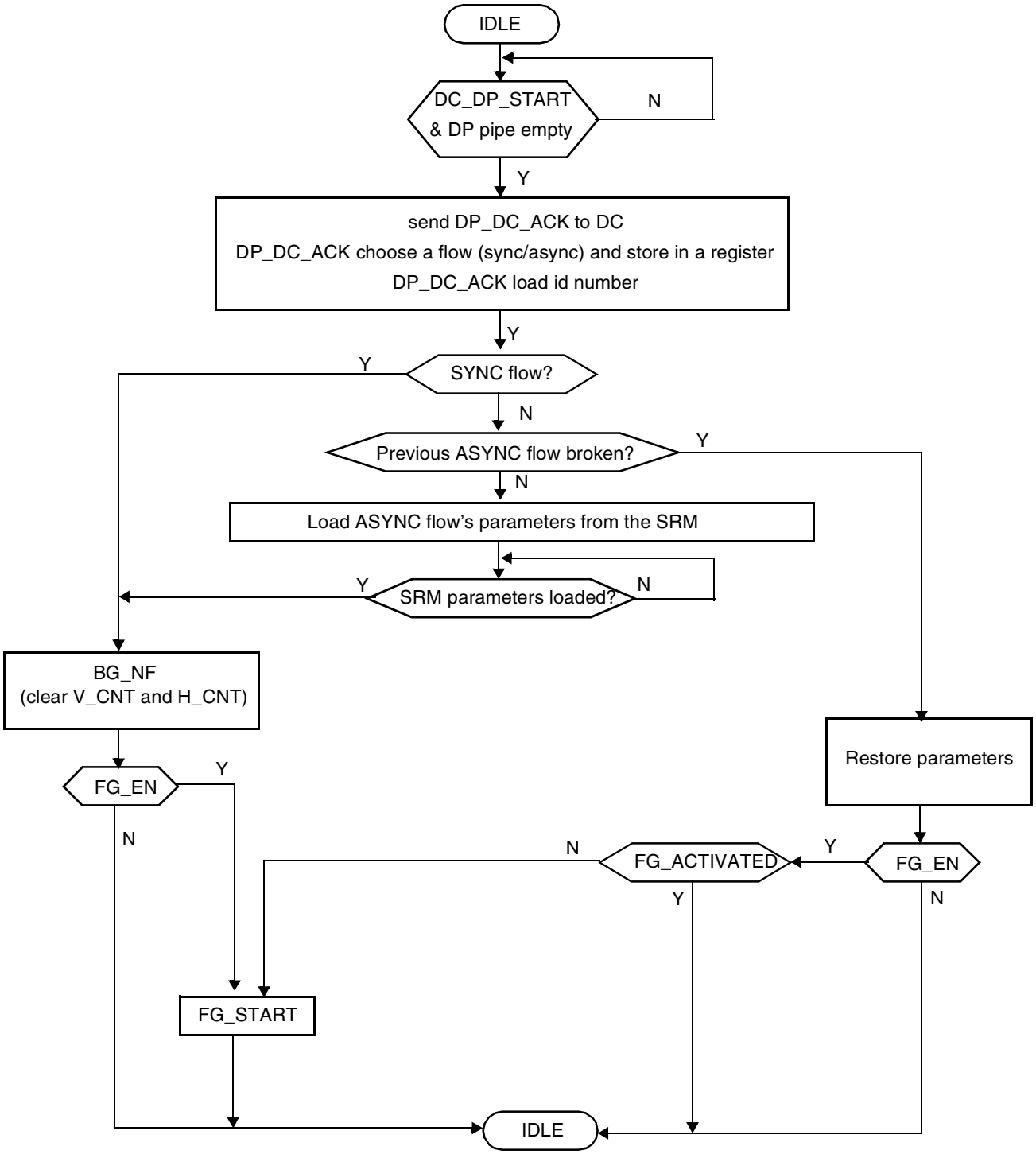


Figure 42-444. DP flow management chart

42.3.9.9 DP debug unit

The DP supports synchronous and asynchronous flows using the same hardware. The asynchronous flow can be broken by the synchronous flow. The DP's debug unit provides the ability to know on which row exactly the async flow has been broken. This is done by providing an interrupt (with DP_DEBUG_CNT register) and providing row status flags (on DP_DEBUG_STAT register).

As the async flow can broken multiple times within a specific frame The user can control the breaking point the issues the debug event by programming BRAKE_CNT field

42.3.9.10 Restriction

When both full and partial planes are processed, the full plane's minimal frame width is 13 pixels.

The Minimum frame height supported by the DP is 2 lines.

42.3.10 Display Interface (DI)

The DI provides arbitrates access to up to three displays with time multiplexing. It converts a data from the DC or the MCU (low level access for serial interface only) to a format suitable for the specific display interface. The DI generates display clocks and other display control signals with programmable timings. The DI outputs data to or inputs from parallel and/or serial interfaces.

This module generates all the control signals sent to the display. The DC sends to the DI; the data for the display and a set of control signals. The controls coming from the DC are used in order to generate the control signals sent to the display. One exception is serial low level access (LLA) where the DC is bypassed and the data is directly coming from the MCU. Figure 42-445 is the DI's block diagram.

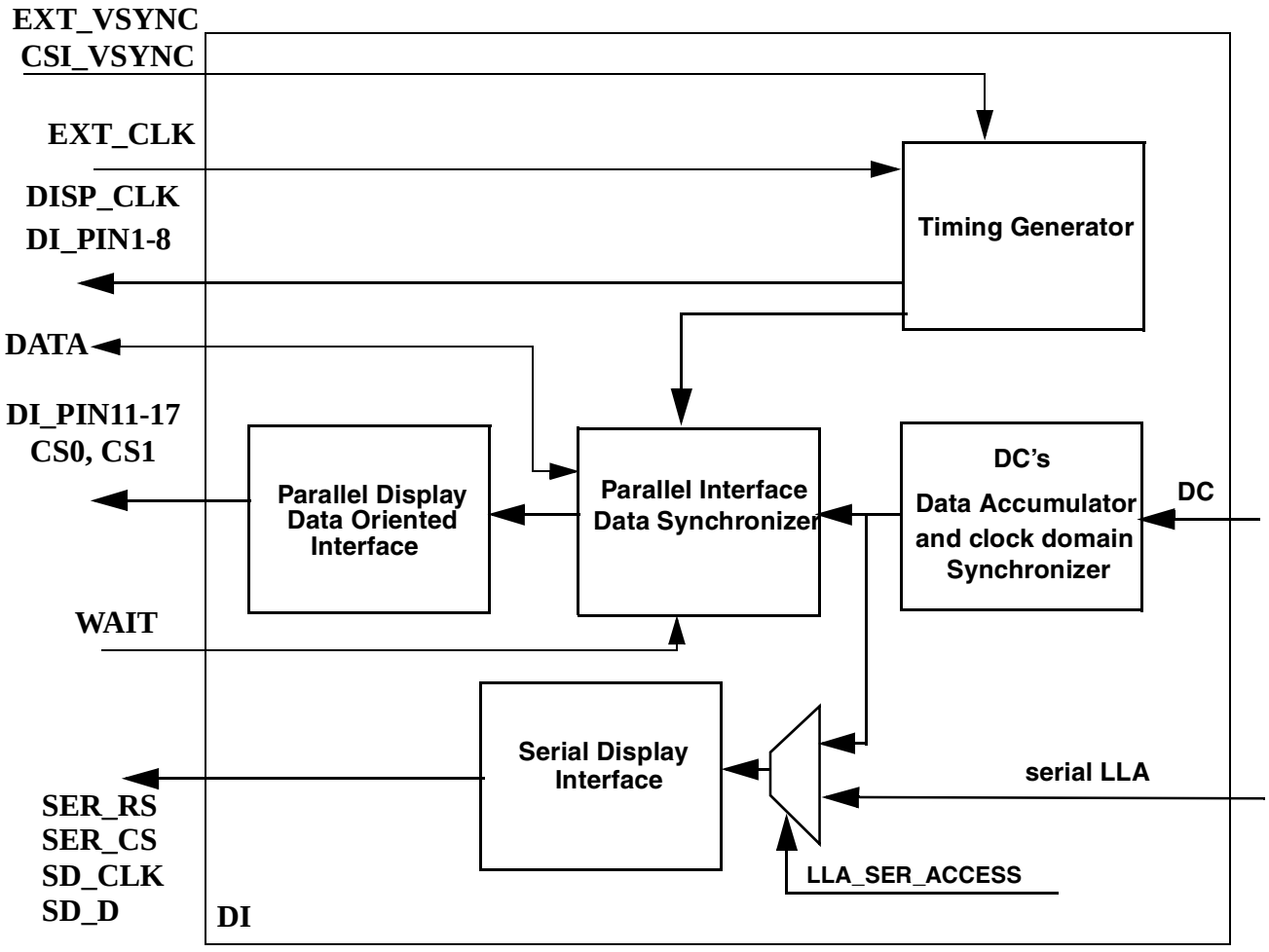


Figure 42-445. DI's block diagram

The display interface includes 2 groups of control signal:

- Time oriented signals - these type of signals are generated according to the DI's internal timers. These are free running signals that change their state according to a pre defined waveform. For example VSYNC, HSYNC, display's clock (pixel clock) etc.
- Data oriented signals - The DC may add markers to data sent to the DI. This markers are used to indicates about a specific attribute of the data (for example: end-of-line, end-of-frame, chip-select etc.). The marker coming from the DC triggers a specific waveform of one or more signals on the display's interface. The specific waveform will be seen on the bus along with the associated data. The markers may be synced to a time oriented signal. For example: attach the end-of-frame signal to the next VSYNC.

42.3.10.1 DC interface, data accumulator and clock domain synchronizer

The data accumulator is the DI's input buffer. It receives the data from the DC along with a set of control signals. The data accumulator receives the data from the DC's clock domain and synchronize it to the DI's clock domain.

42.3.10.2 Parallel interface data synchronizer and data oriented interface

The data accumulated in the DI's input buffer will be sent to the display according to the DI's internal events. Each event is generated by a counter. A tag is attached to each data by the DC's microcode using the SYNC field in the DC's microcode. The tag selects the event that the data will be synced to. Once the corresponding event is generated the data will be sent to the display. A data can be a pixel or a component (part of a pixel).The data synchronization occurs separately for each component. For asynchronous displays the tag will be equal to 0 i.e. the data is not synchronized to any event. The data will be sent out of the buffer immediately.

42.3.10.3 Timing generator

The timing generator is used for generating the waveforms' of each pin of the display's interface.

The timing generator is built of 10 counters. One counter functions as a time base This timer is called the BASE TIMER (BS). The other 9 counters are used in order to generate the control signals' waveforms. The last counter (counter #9) is special see Section 42.3.10.3.3, "Counter #9"

The DI clock can be derived from IPUv3EX's clock (HSP_CLK) or from an external source (via the ipp_di_#_ext_clk pin). The clock's source is statically selected by configuring the DI#_CLK_EXT bit.

Figure 42-447 illustrates the main parameters of a waveform. A waveform's segment is built of 5 parameters. Each segment can be has 2 phase "ACTIVE PHASE" and "OFFSET PHASE".

- TIMEBASE - this is the base timer (fundamental timebase); all the other parameters are derived from this timer. The timebase is generated by counting DI clock cycles. The amount of cycles is defined according to DI#_DISP_CLK_PERIOD. This field defines the Display interface clock period, This parameter contains an integer part (bits 11:4) and a fractional part (bits 3:0). It defines a fractional division ratio of the Di's source clock for generation of the display's interface clock. The timebase is generated from edge aligned pulses of the DI clock.
  The user can delay the timebase starting point by defining an offset to the timebase. This is done by configuring the DI#_DISP_CLK_OFFSET field. The offset is calculated from the point where the DI is enabled to the point where the timebase starts ticking. The display clock waveform is generated between 2 edges of the timebase. The waveform is defined according to the DI#_DISP_CLK_UP and DI#_DISP_CLK_DOWN. Each pin has a specific timebase that is derived from the fundamental timebase
  Figure 42-446 illustrates the relations between the TIMEBASE and the DI's clock

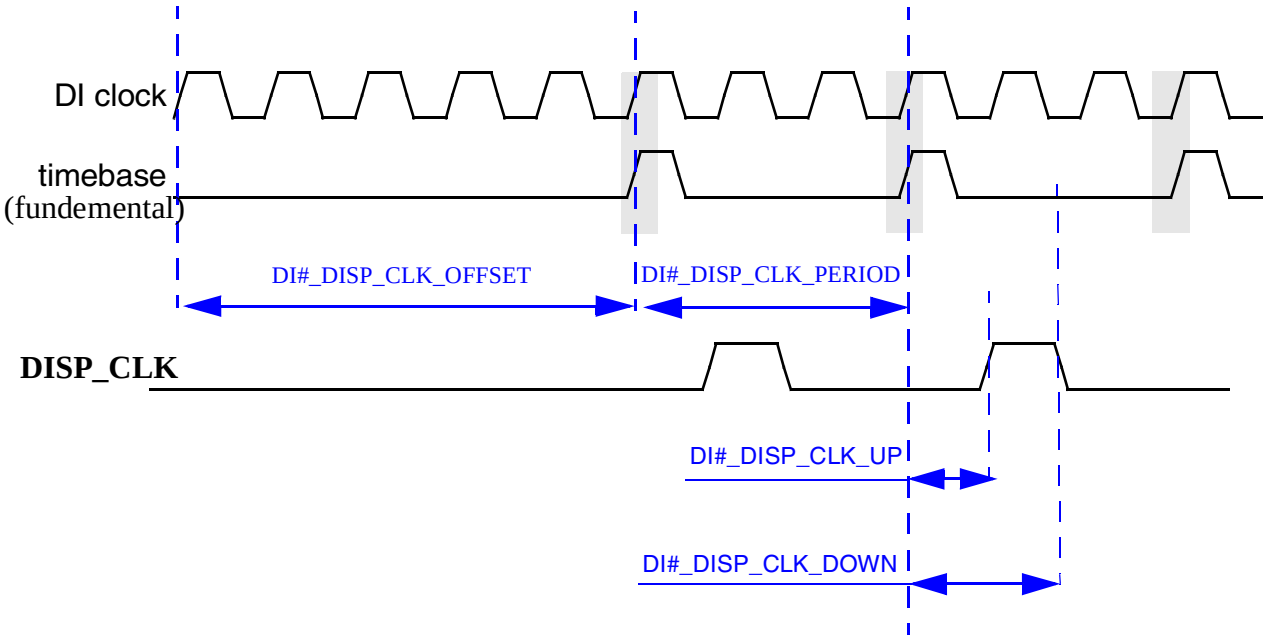


Figure 42-446. Timebase, DI's clock and display's clock relations

- OFFSET - this parameter defines when the length of the "OFFSET PHASE" it is defined by di#_offset_value_<N> field, where N is the counter's index.
- STEP - This parameter defines the length of the "ACTIVE PHASE"; it is defined by the di#_step_repeat_<N> field, where N is the counter's index. If the counter is in auto reload mode (di#_cnt_auto_reload_<N> bit is set) then the counter will be automatically reloaded forever. The value of di#_step_repeat_<N> is ignored in that case.
- RUN - The "ACTIVE PHASE" is partitioned to several "RUN sections"; this parameter defines the length of the "RUN section"; it is defined by the di#_run_value_m1_<N> field, where N is the counter's index.
- UP - Each "RUN section" contains the waveform of a single pulse. This parameter defines the offset from the beginning of the "RUN section" to the assertion of the signal; it is defined by the di#_cnt_up_<N> field, where N is the counter's index.
- DOWN - This parameter defines the offset from the beginning of the "RUN section" to the negation of the signal; it is defined by the di#_cnt_down_<N> field, where N is the counter's index. In case where DOWN < UP the waveform will have a 50% duty cycle.

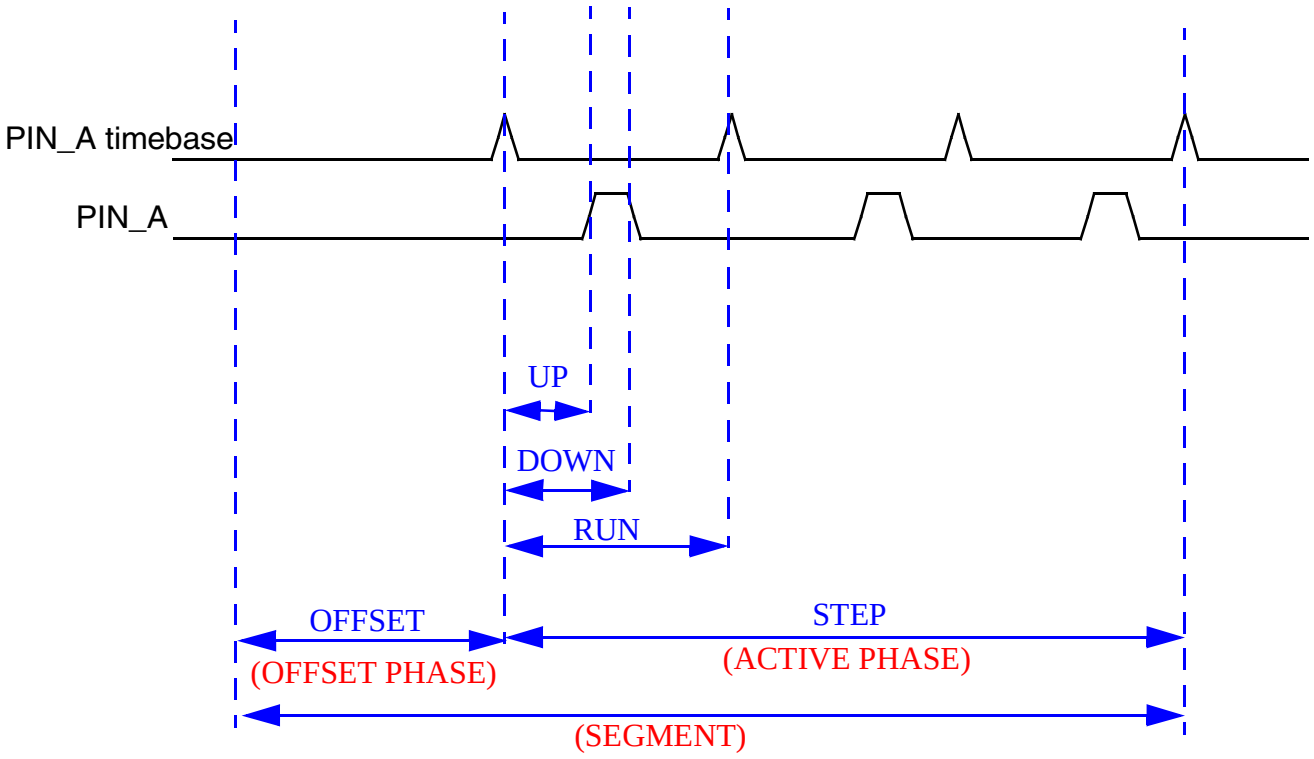


Figure 42-447. DI waveform's main parameters

42.3.10.3.1 Waveform's concatenation

The DI provides the ability to derive the waveform from the fundamental timebase or from another PIN. In that case one pin's waveform is used as another pin's timebase. Figure 42-448 is an example. PIN_A and PIN_C are derived from the fundamental timebase however PIN_B is derived from PIN_A's wavefore.The trigger is selected by DI#_RUN_RESOLUTION_<N>, where <N> is the index of the counter. A counter can be triggered by counter with lowered index. For example: counter #5 can be triggered by counter #3 but can't be triggered by counter #7.

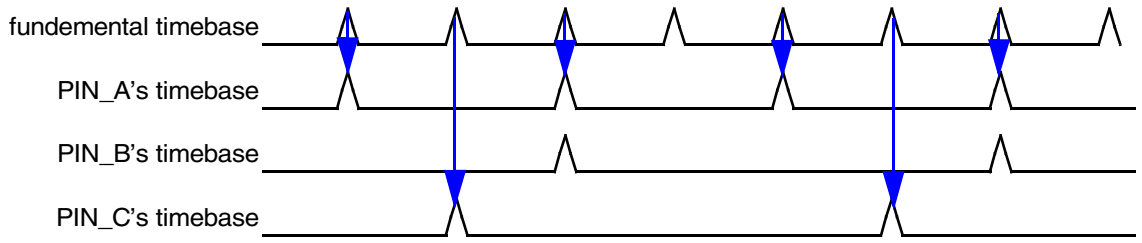


Figure 42-448. DI pins - Waveform's time bases concatenation

42.3.10.3.2 The basic counter

The DI has 9 counters. Figure 42-449 illustrates the counter's structure. A counter is built of 3 units: timebase generator, waveform generator and polarity generator.

The timebase generator

The timebase generator gets 3 triggers. Clear, offset and run trigger. The Clear is the trigger that resets the counter. It is selected by programming the DI#_CNT_CLR_SEL_<N>.

The Offset trigger is the trigger used for counting the OFFSET_PHASE. The user can select the source of the trigger by programming the DI#_OFFSET_RESOLUTION_<N>. The offset's value is defined by programming the DI#_OFFSET_VALUE_<N>

The RUN trigger is the trigger used for counting the RUN period. The user can select the source of the trigger by programming the DI#_RUN_RESOLUTION_<N>. The RUN's value is defined by programming the DI#_RUN_VALUE_<N>

The timebase generator counts according to the DI_CLK or according to another counter's output. In order to use a source different than DI_CLK, the user should set the POLARITY_GEN_EN bits to 01.

Waveform generator

This unit generates the waveform. It gets the values of RUN, UP, DOWN and STEP and build the waveform accordingly. The waveform is counted according to the signal generated by the timebase generator.

Polarity generator

The waveform's polarity is controlled by 2 units. The static polarity is changed according to the POLARITY_<#> bit of each pin. This bit defines if the waveform of the pin is active high or active low.

The other unit controlling the polarity is the polarity generator. This unit is enabled by setting the POLARITY_GEN_EN[1] to 1. The polarity generator has 2 modes: “toggle mode” and “normal polarity mode”. The mode is defined according to POLARITY_GEN_EN[0].

- Normal polarity mode - In this mode the polarity is changed according to 2 other counters. The counter selected by POLARITY_TRIGGER_SEL define the sampling point. The counter selected by POLARITY_CLR_SEL defines the polarity value. At the sampling point the polarity value is defined. The current polarity value defines the current polarity of the waveform.
- Toggle mode - In this mode, the output of the timebase generator’s output causes the polarity to toggle. Any tick of the timebase generator inverts the polarity. When this mode is enabled the ticks generated by the counter selected by POLARITY_TRIGGER_SEL initialize the polarity generator and the timebase generator causes the polarity to toggle.

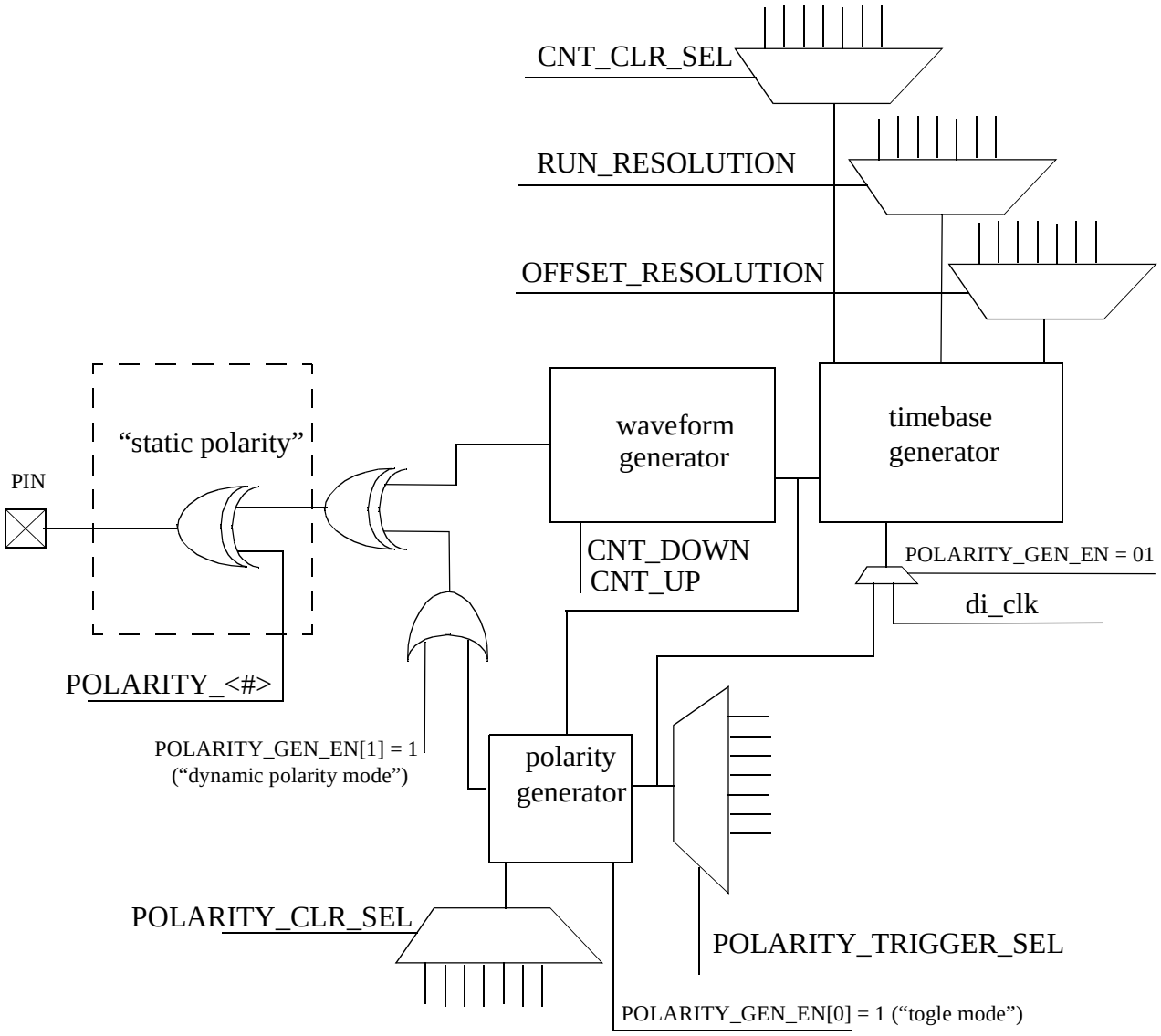


Figure 42-449. DI's counter's structure

Figure 42-450 provides an example for waveform generation using different trigger sources. The waveform is generated for PIN_D. PIN_D counter is cleared by the PIN_C's timebase. The offset period is calculated by counting cycles of PIN_B's timebase. The RUN period is calculated by counting cycles of PIN_A's timebase. The UP and DOWN periods of the waveform are calculated by counting cycles of PIN_D's timebase.

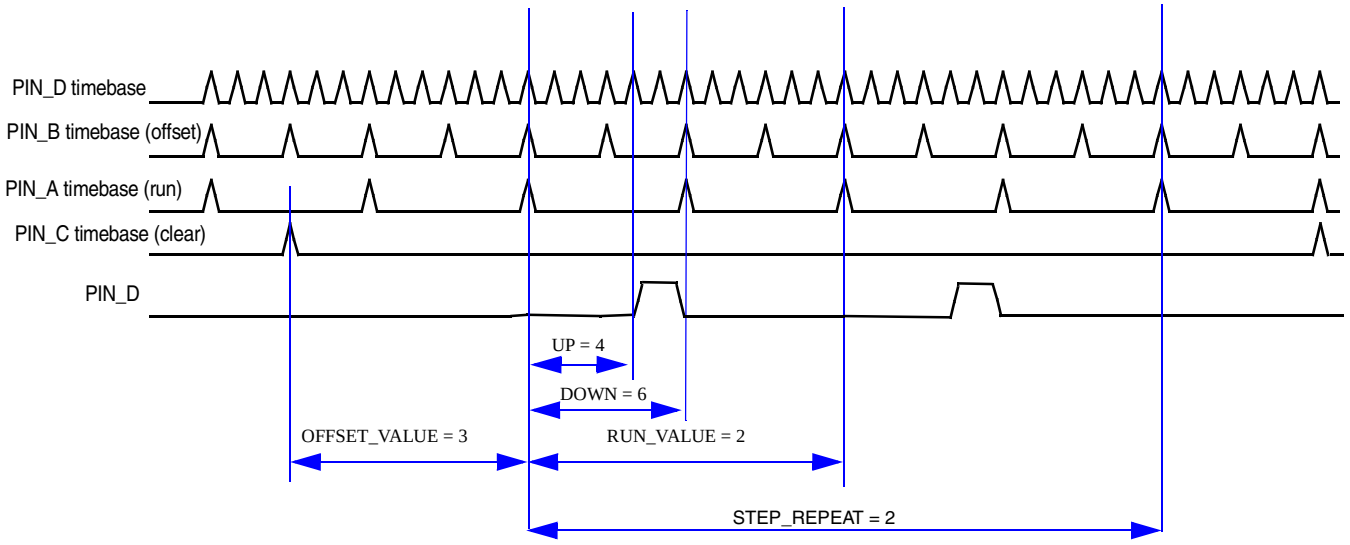


Figure 42-450. Clear, offset and run triggers and values - Example

42.3.10.3.3 Counter #9

The last counter (counter #9) is an auxiliary counter and it is not used for generating a waveform for a specific pin. It can be used in order to attach another waveform to an existing one. The user define the waveform for a specific pin (main pin). The user defines the auxiliary waveform using counter #9. The 2 waveforms are logically ORed. The combined waveform will be routed to the main pin. The main waveform that this counter is attached to is defined according to DI#_GENTIME_SEL_9.

The tag of counter #9 can be generated from counter #9 or from the main waveform. This is selected according to the DI#_TAG_SEL_9.

42.3.10.3.4 DI's active window

The DI provides an alternative way to define the synchronous display setting by using an active window and thus need to program less counters. The synchronous display's active window is a rectangle on the display where IPUv3EX sends data. The active window is set by programming the parameters defined on DI#_AW0 and DI#_AW1 registers. Figure 42-451 illustrates the different parameters defining the active window. The display's vertical and horizontal position are defined according to counters. The DI#_AW_HCOUNT_SEL selects the counter which the horizontal position is calculated according to. DI#_AW_VCOUNT_SEL selects the counter which the vertical position is calculated according to. The Active data is sent according to a trigger. DI#_AW_TRIG_SEL selects the counter which calculates this trigger. This trigger usually functions as a data enable signal.

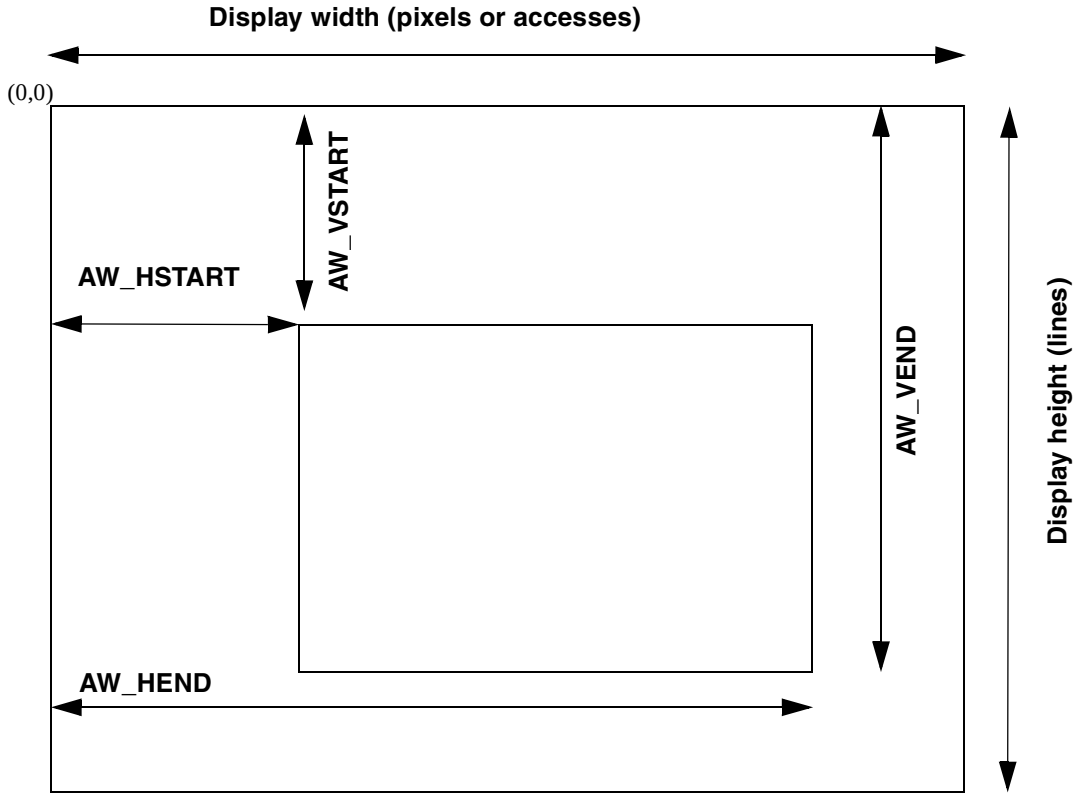


Figure 42-451. DI's Active Window

42.3.10.4 Waveform settings for asynchronous interface pins

The DI provides 8 signals that are used for asynchronous interface. These signals are PIN11 through PIN17 (ipp_di_#_pin_11 through ipp_di_0_pin_17) and the CS (ipp_di_0_do_dispb_d0_cs). The DI holds 12 wave set quartets. Each quartet includes 4 registers (DI#_DW_SET<j>_<i>). Each DW_SET register holds the UP and DOWN values of the waveform. The DW_SET register is selected in the following way. The WAVEFORM field in the DC template is a pointer that points to one of the 12 quartets. In addition the WAVEFORM field points to one of 12 DI#_DW_GEN_<i> registers. The DI#_DW_GEN_<i> holds 9 pointers. The pointers are 2 bits field that points to one of the registers from the DI#_DW_SET<j>_<i> quartet. Each one of the 8 pointers in the DI#_DW_GEN_<i> registers is related to a specific pin of the DI's asynchronous interface. Figure 42-452 illustrates the relations between the registers controlling the asynchronous interface's signals

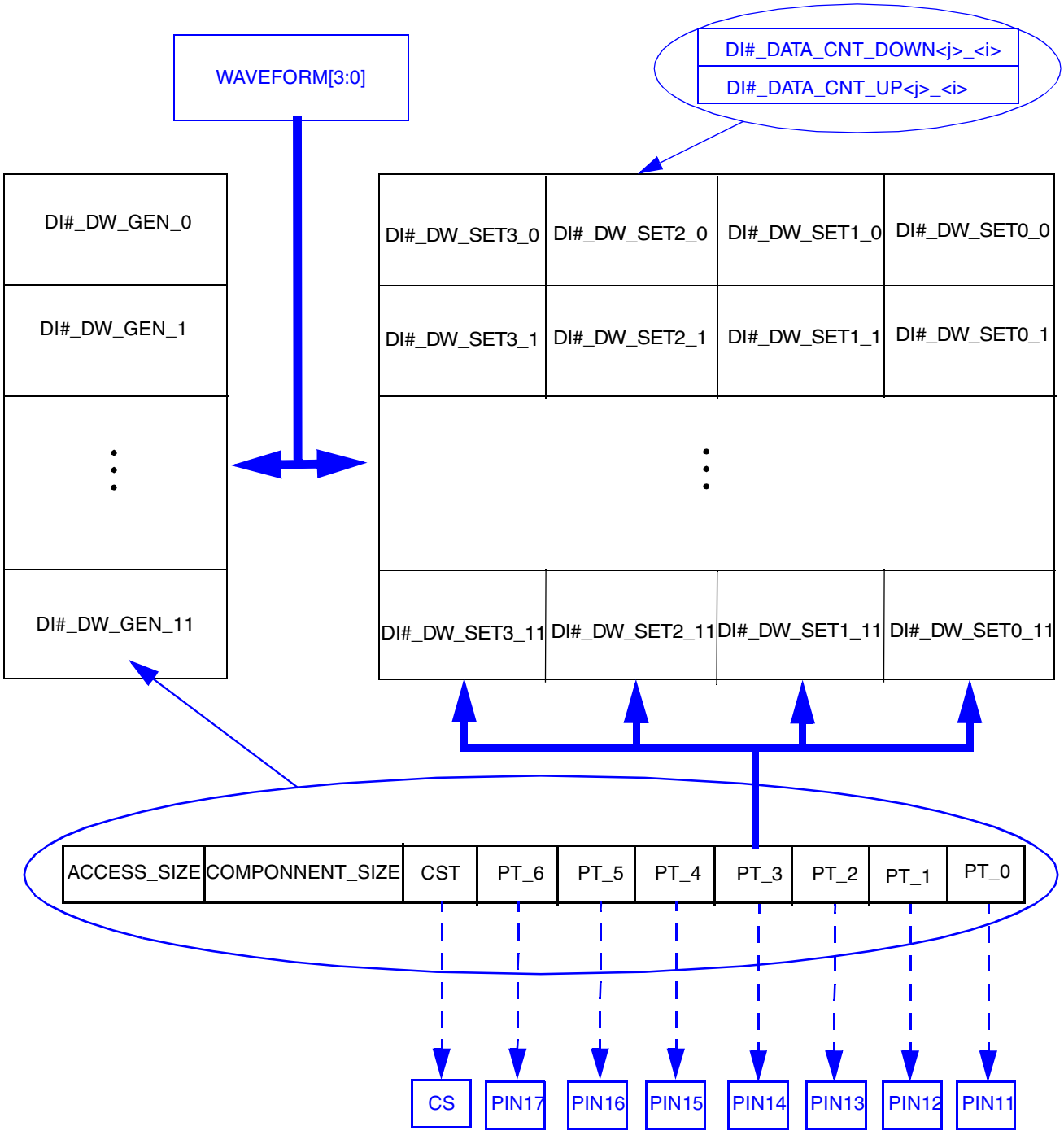


Figure 42-452. Waveform settings for asynchronous interface pins - parallel interface

The DI#_DW_GEN_<i> register includes the data's waveform settings as well. ACCESS_SIZE defines the amount of DI clock cycles that a pixel is valid on the bus. When generic data is sent, this field defines

the amount of cycles that the generic data is valid on the bus. A pixel may be broken into few components. The COMPONNENT_SIZE field defines the amount of cycles that each component is valid on the bus.

The COMPONNENT_SIZE is always smaller or equal to ACCESS_SIZE. For synchronous interface COMPONNENT_SIZE is always equal to ACCESS_SIZE. In case that there's a need for some gap between one type of accesses to another having the COMPONNENT_SIZE smaller than ACCESS_SIZE can be useful (for example read after write accesses that require some gap between them).

42.3.10.5 Serial display interface

The DI supports the following asynchronous serial interfaces:

1. 3-wire (with bidirectional data line).
2. 4-wire (with separate data input and output lines).
3. 5-wire type 1 (with sampling RS by the serial clock).
4. 5-wire type 2 (with sampling RS by the chip select signal).

- For serial interfaces, accessed in LLA mode, data and preamble lengths and other parameters are controlled via the DI0_SER_CONF and DI1_SER_CONF Registers.

The serial display interface includes generators for the CS,RS and CLK signals of the display interface and a bidirectional datashifter. Figure 42-453 provides a block diagram of the serial display interface.

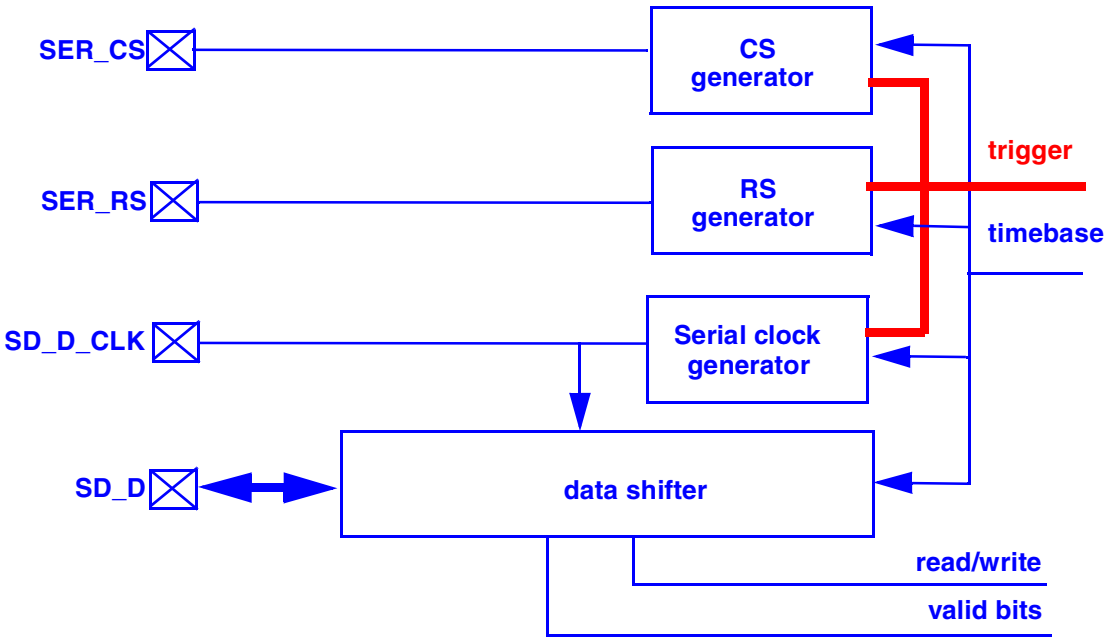


Figure 42-453. Serial display interface

42.3.10.5.1 Waveform settings for serial interface pins

When serial interface is used the fields in the DI#_DW_GEN_<i> register have different meaning than the parallel interface. The registers includes pointers to DW_SET for the CS, RS and CLK pins of the serial interface. Figure 42-454 illustrates the relations between the registers controlling the serial interface's signals

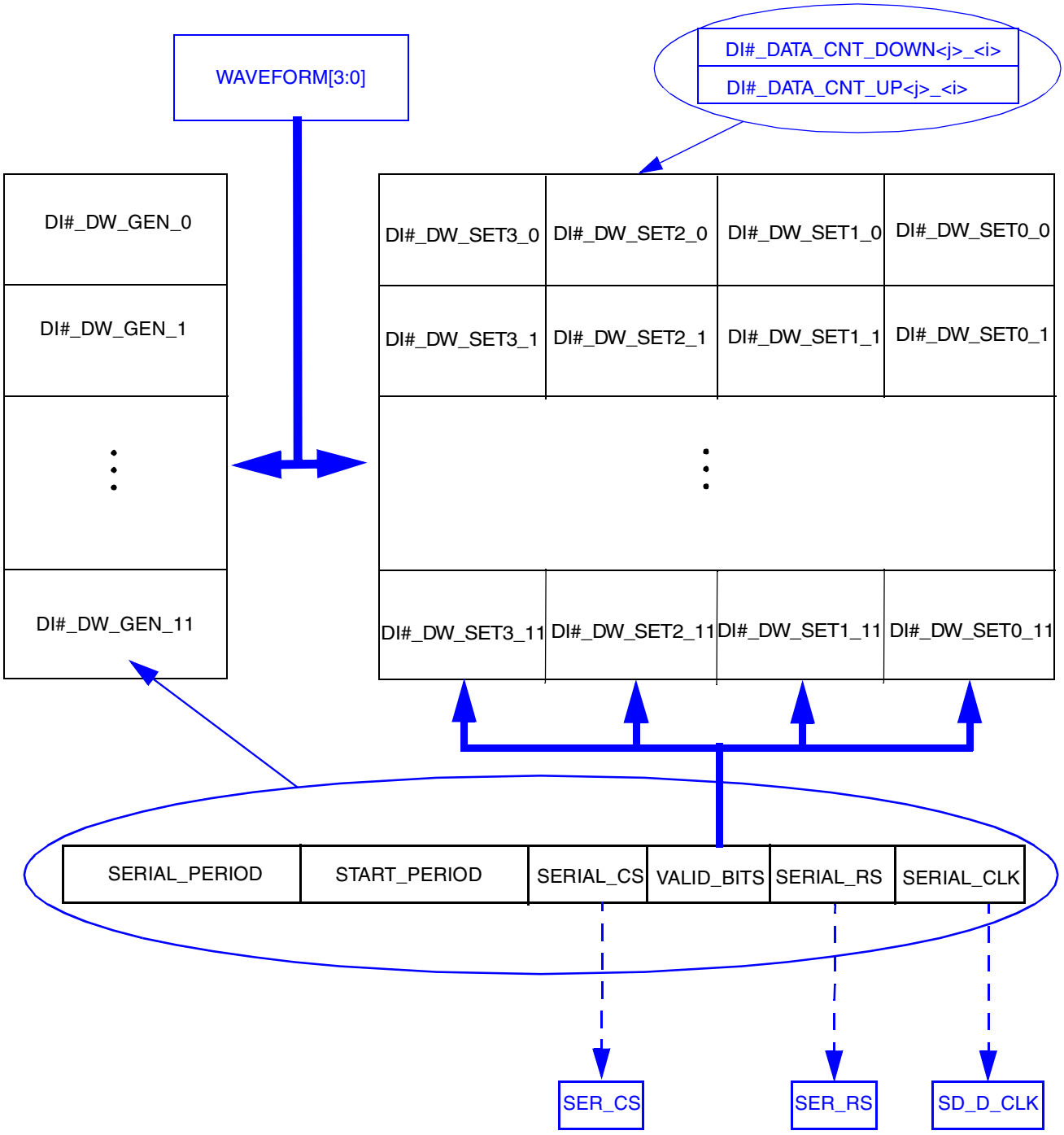


Figure 42-454. Waveform settings for asynchronous interface pins - serial interface

Figure 42-455 provides an example of the waveform's settings for the serial interface. A waveform generation starts following a trigger indicating that the serial interface has won the DI's arbitration. Therefore the next access will be via the serial interface. The waveform is based on timebase ticks. All the

waveform's settings are defined by counting DI_CLK cycles. The timebase starts ticking after counting DI_CLK cycles defined on the START_PERIOD parameters. The number of timebase ticks is defined according to the VALID_BITS field. The VALID_BITS field defines how many bits will from the 32 bit word sent to the DI are valid and should be sent via the serial interface. The valid bits are the least significant bits of the word coming from the DC. The amount of DI_CLK cycles between each timebase tick is defined according to the SERIAL_PERIOD parameter. Each one of the serial interfaces control signals may have a different waveform defined according to its corresponding UP and DOWN values.

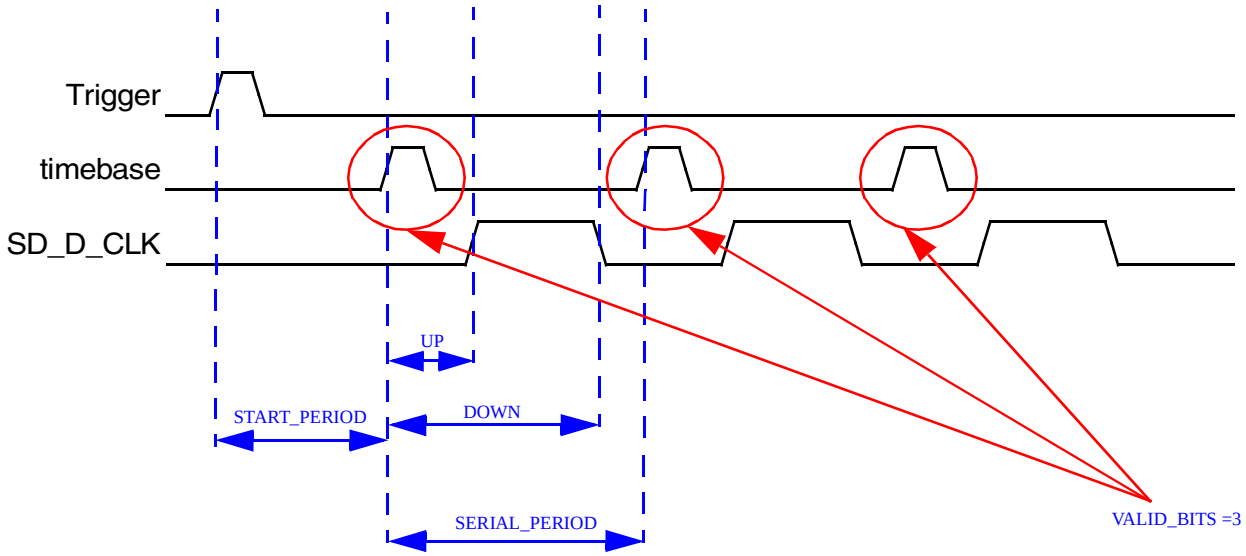


Figure 42-455. Serial interface waveform example

The CS and RS signals can have a single change (one rising and one falling edges) between 2 consecutive triggers.The UP value for RS and CS is measured from the first timebase tick. The DOWN value is measured from the last timebase tick. Figure 42-456 provides an example of the RS behavior, CS behaves in a similar way.

The data is sent according to the timebase ticks. The MSB bit of the data is transmitted first. When receiving serial data, the data is sampled every timebase tick. The user can program an offset from the timebase tick by programming the DI#_SERIAL_LATCH field. This value will move the serial data sampling point. The value is measured by counting DI_CLK cycles.

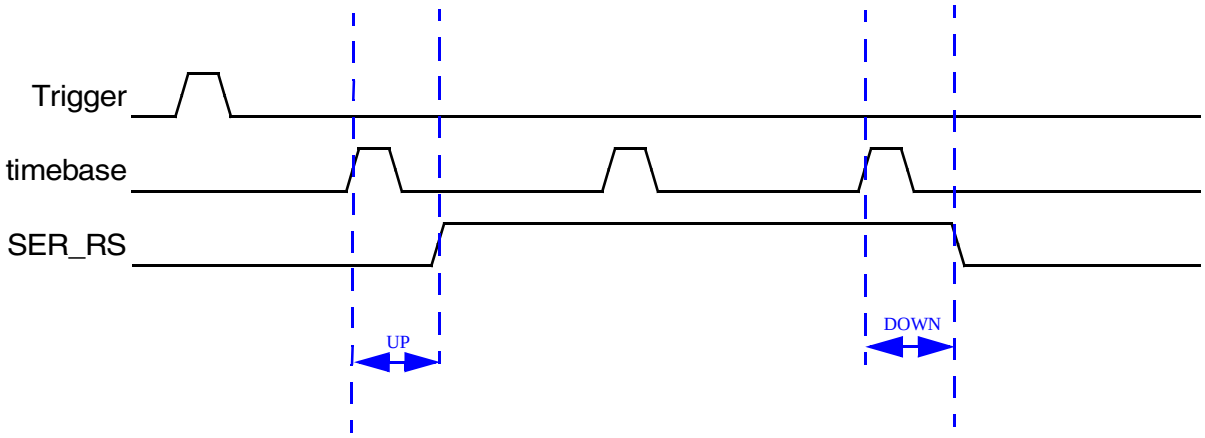


Figure 42-456. Serial interface waveform example of RS

42.3.10.6 Low Level Access - LLA

LLA is a MCU direct access to the display. For parallel displays the DI's behavior for LLA is the same as any other access to a parallel display. For serial displays the DI allows a DI arbitration bypass. This is done if the DI#_LLA_SER_ACCESS bit is set. IN that case there is a direct access from the DMFC to the DI which bypasses the DC allowing simultaneous access to both serial and parallel interfaces. When DI#_LLA_SER_ACCESS bit is set, the user must not do any other kind of accesses to the serial interface except LLA. When DI#_LLA_SER_ACCESS bit is set the corresponding DI#_WAIT4SERIAL must be clear.

42.3.10.7 Using a mask channel

The IPUv3EX is able to provide the windowing function on displays that have data enable control. This is achieved by masking of some screen regions according to a 1-bit/pixel mask read from the memory via IDMAC through channel #44. When the mask value is zero, the pixel is not displayed. This feature can be used for dual-port smart displays.

42.3.11 Video De Interlacing Module (VDI)

The Video De-interlace block (VDI) deinterlaces standard interlaced video to produce progressive video, that is used for upsizing to HD formats or for display on progressive displays. For VDI operation three fields are necessary F(n-1),F(n),F(n+1). The F(n-1) field arrived through CSI interface in real time mode or through channel 1 and then stored in FIFO1. The F(n) arrived through channel 2. At least three lines of F(n) are stored in Line Store memory. The F(n+1) arrived through channel 3 and stored in FIFO3. FIFO controllers read data from FIFOs and then data aligned in pixels buffers. From the buffers the data arrived to Line Padding Controller (LPC). The LPC padding missing line at the beginning and end of the frame.

The DeInterlacing module (DI) perform the processing. Then data send to IC module for processing or for transferring to external memory.

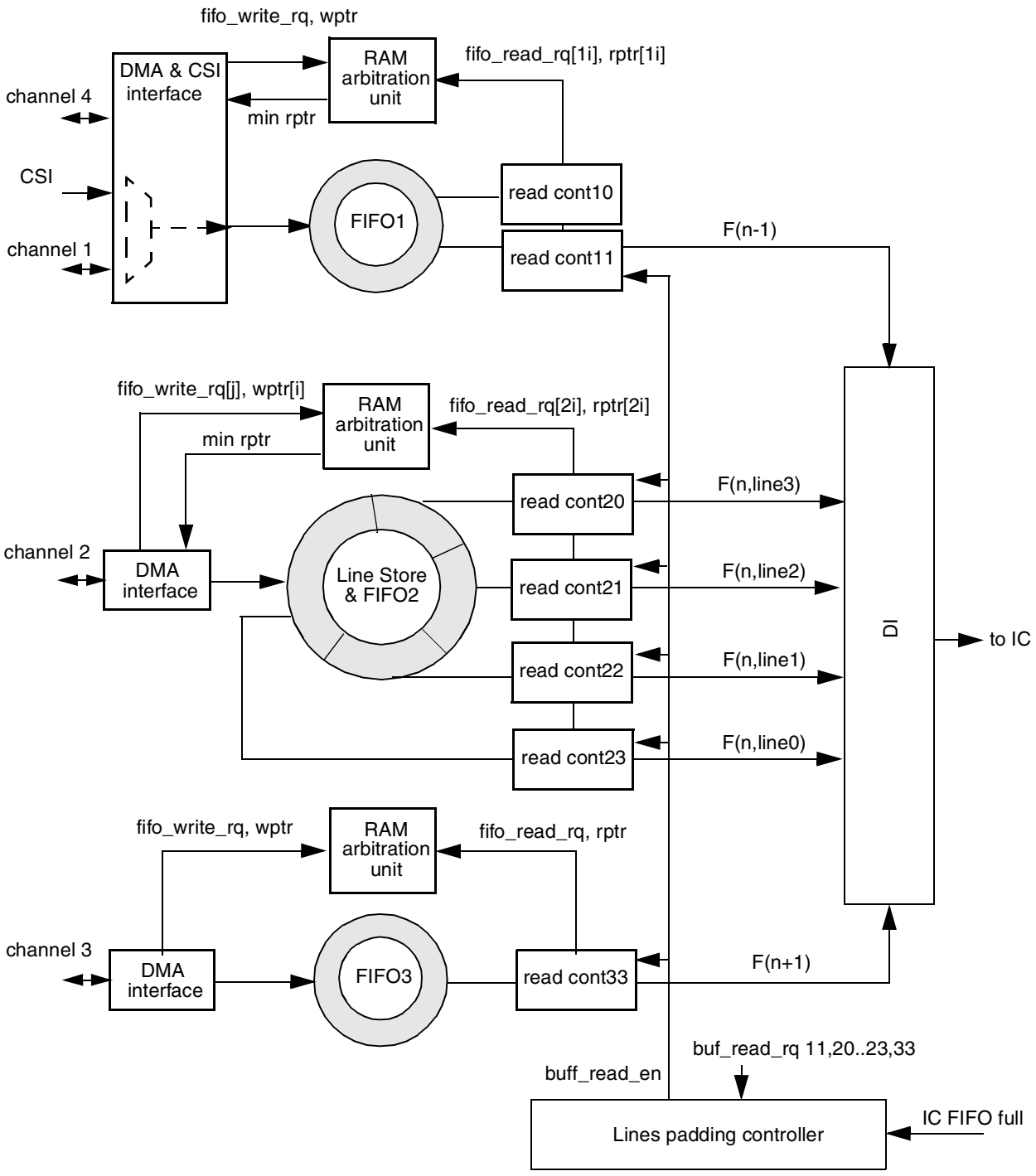


Figure 42-457. VDI Block Diagram

42.3.11.1 VDI Features

Key features of the VDI module include:

- Deintarlacing
  - maximum horizontal resolution 720 pixels
  - maximum pixel rate 60MP
  - Support YUV422 and YUV420 formats
- CSI FIFO mode

42.3.11.2 De interlacer (DI) module

The block diagram of the DI block is shown in **Figure 42-458.** Pipelining is inserted at all stages, so that the design may run at a fast clock speed, if needed.

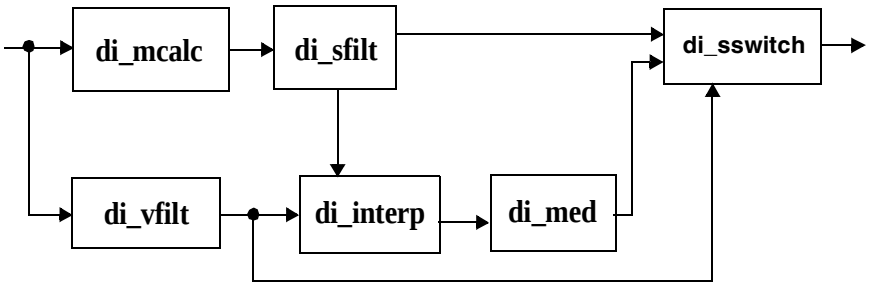


Figure 42-458. DI block diagram

42.3.11.2.1 Vertical Filter Block (di_vfilt)

The di_vfilt block performs spatial vertical filtering of pixels. It is a four tap vertical filter:

$$vfilt\_out = (-3.0*pix1 + 19.0*pix2 + 19.0*pix3 - 3.0*pix4)/32.0 \qquad \textit{Eqn. 42-11}$$

Where pix1, pix2, pix3, pix4 are four pixels in same horizontal location on four consecutive lines of a field. vfilt_out is pixel being predicted (between pix2 and pix3)

42.3.11.2.2 Motion Calculator Block (di_mcalc)

The mcalc block estimates the amount of motion for any given pixel by looking at pixel values in current, previous and next fields. The generic formula used to calculate the estimated motion is:

$$m = SAT(Ks*(|e-w|/(|e-w|+|n-s|+SPA\_DETAIL))) \quad \textit{Eqn. 42-12}$$

- Where, n is the pixel above the pixel being predicted
- s is the pixel below the pixel being predicted
- e is the pixel in previous field at same spatial location as the pixel being predicted
- w is the pixel in next field at same spatial location as the pixel being predicted
- m is motion estimation for the pixel being predicted (range from 0 to 1)
- Ks is slope control and decides how quickly the algorithm switches from no motion (m=0) to full motion (m=1)
- SPA_DETAIL is a constant (50) that is added to |n-s|.
- SAT() is saturate at 1 function

The motion calculator block is simplified in a certain respect, by removing the need for a divider, while providing a degree of flexibility. The main motivators for this are the following observations:

1. the above equation defines a set of curves based on the value of |n-s|, but the curves are fairly closely spaced, so that using a granularity of 8 in |n-s| to define specific curves to use gives pretty much the same quality of picture as using all of the curves; and once |n-s| reaches about 120, the effect of an increased |n-s| value is hard to observe
2. once |e-w| gets to about 15, the motion is usually saturated to a value of 1

Based on the above observations, the motion calculator is now implemented with the use of two “ROM”s, using the 4 LSBs of |e-w| and bits 6:3 of |n-s|.

The motion calculator has 3 modes of operation. These modes are defined by the VDI_MOT_SEL field. In case that the user has an information about the motion (For example SW analyzing motion vectors provided by a video decoder) he can select one of the modes listed below. Changing the value of the VDI_MOT_SEL has an affect only on the next frame.

- When VDI_MOT_SEL==2'b01, m_calc is 0 (no motion – use weave)
- When VDI_MOT_SEL==2'b10, We assume high motion and use min(15,delta_t).
- When VDI_MOT_SEL==2'b00, We assume low motion and use sat([0,15],delta_t-8) (delta_t-8 is signed operation).

42.3.11.2.3 Spatial Motion Filter (di_sfilt)

The di_sfilt block spreads motion signal over five pixels:

$$M_{spread} = MAX(m3, (0.5*m1 + m2 + m3 + m4 + 0.5*m5)/4.0) \quad \textit{Eqn. 42-13}$$

- Where, m3 is motion estimate for current pixel
- m2 is motion estimate for previous pixel
- m4 is motion estimate for next pixel
- m1 is motion estimate for pixel before previous pixel
- m5 is motion estimate for pixel after next pixel

42.3.11.2.4 Interpolated Pixel Calculator Block (di_interp)

The di_interp block uses the motion estimated by the di_sfilt block and computes an interpolated pixel that is weighted sum of the surrounding four pixels (n, s, e, w). The block performs the following calculations:

```
        if (M_spread <= 0.5) {
i = (1-2*M_spread)*(e+w)/2 + 2*M_spread*vfilt_out
} else {
        i = vfilt_out
}
```

Where,i is the interpolated pixel

n, s, e, w are surrounding pixels as explained earlier

42.3.11.2.5 Median Filter Block (di_med)

The di_med block performs a 5-point median of n, s, e, w and i pixels. It should be noted that the median required here is not a true 5-point, but can be implemented more efficiently as a 3-point median:

$$\text{med} = \text{MEDIAN}(\min(\max(n,s),\max(e,w)), \max(\min(n,s),\min(e,w)), i) \quad \textit{Eqn. 42-14}$$

42.3.11.2.6 Soft Switch Block (di_sswitch)

The final output of the deinterlacer is a blend of the median value and the vertical filter, assuming that the pixel data n,s are uncorrelated with the pixel data e,w. By uncorrelated, we mean (max(n,s) < min(e,w)) or (min(n,s) > max(e,w)).

```
        if (M_spread <= 0.5 || not(uncorrelated)) {
pix_out = med
} else { /* M_spread > 0.5 */
pix_out = (1-2*(M_spread-0.5))*med + 2*(M_spread-0.5)*vfilt_out
}
```

42.3.11.3 DMA only Mode

In DMA only mode the data is coming from IDMAC only.

42.3.11.4 Real Time Mode

In Real Time Mode the F(n-1) are coming from CSI. The CSI write to FIFO1. The DI module read F(n-1) from processing. In addition IDMAC read the field from FIFO1 and store in external memory. Then stored frames are used as F(n) and F(n+1).

42.3.11.5 CSI only Mode

In CSI only mode VDI do not perform any processing and used as FIFO only. The data arrived from CSI written to FIFO1. The IDMAC read data from the FIFO1. The FIFO3 and Line store memory are not used. The ID module are turned OFF. The CSI only mode can be used as alternative to SMFC, when appropriate.

42.3.11.6 VDI Restrictions

- Maximum output pixel rate is 60MP.
- The output pixel format (YUV422 or YUV420) are equal to input format.

42.3.12 Control Module (CM)

42.3.12.1 Block Diagram

The CM Block Diagram is shown in Figure 42-459.

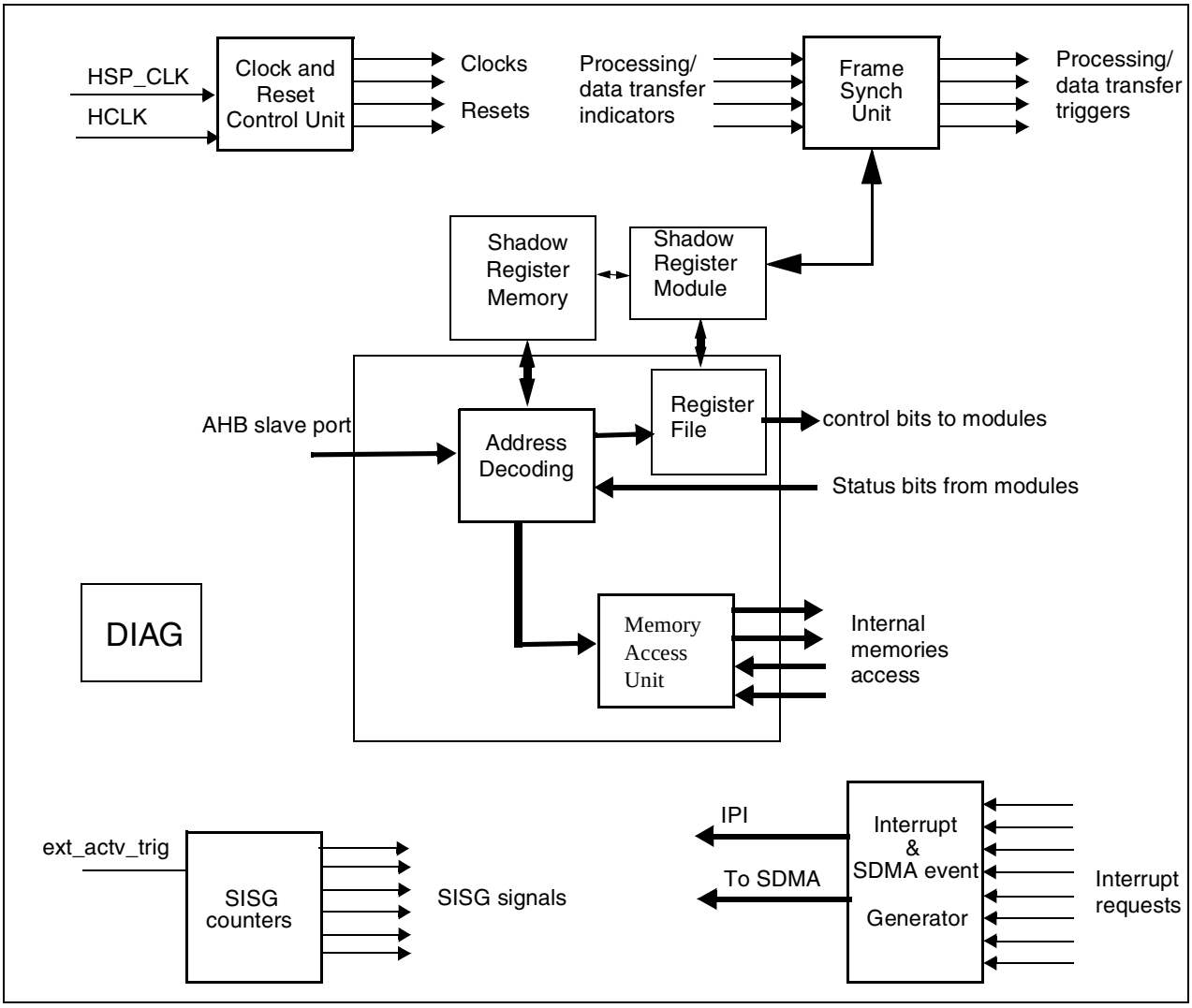


Figure 42-459. CM Block Diagram

The CM consists of the Frame Synchronization Unit (FSU), the Interrupt Generator (IG), the General Configuration Registers (GCR), the Clock and Reset Control Unit (CRCU) and the Shadow Registers Module (SRM).

42.3.12.2 Frame Synchronization Unit

42.3.12.2.1 General Description

The FSU provides synchronization of tasks performed by different IPUv3EX sub modules and MCU tasks. This allows to build complex processing flows which are performed automatically (without MCU involvement in synchronization of the IPUv3EX's tasks). The FSU supports double buffering of image frames stored in the external memory and allows chaining IPUv3EX processing flows in automatic mode. Table 42-430 describes the most important use cases of chaining the IPUv3EX tasks.

42.3.12.2.2 Frame Synchronization Flow

1. Initialization

   The MCU initializes all the parameters for a task by writing to the GCR and to the parameters memories of each IPUv3EX sub-modules. The initialization must occur before the MCU enables the task.

2. Enabling

   After the initialization step has been completed, the MCU enables the task by setting its enable bit in an appropriate register.

3. Triggering

   After the task is enabled, the FSU waits for triggering signal. The triggering signals is a combination of the enable bit and the buffer ready signal which can be driven by the MCU (DMA_CH_BUF<0/1>_RDY_<#>) and/or by the preceding task (IDMAC_EOF_# or Frame Complete signal from the DC).

   The trigger causes the FSU to invoke the relevant unit to start by assertion of the NEW_FRM_RDY signal. In some cases triggering occurs at the enabling step (when the enable bit is the trigger for the task).

4. Operating

   The triggering step cause the task to move to active mode, this is the operating step. In this step, the FSU monitors the synchronization signals from MCU, IDMAC and the corresponding processing units, and controls the units operation. The FSU also controls the IDMAC buffer toggling when double buffer page flipping is used.

   The FSU checks at end of each frame if the next frame can be served. If the answer is yes, the FSU stays in active mode with re-sending the <TASK>_NEW_FRM_RDY signal and updating the relevant flags (e.g. DMA<BL>_<#>_CUR_BUF and DMA<BL>_<#>_BUF_RDY). If the answer is no, the FSU moves to pause mode and pauses the task waiting until the next frame can be served.

5. Disabling

   When the task is disabled by the MCU (by negating the enable bit), it moves back to non-active mode.

42.3.12.2.3 FSU's fundamentals

Trigger source select

A flow is triggered by a trigger. The trigger may be asserted manually by the MCU or may be a result of the completion of the preceding task. Trigger's source select choose the source of the trigger. The trigger means that the data is ready to be processed by the sub module. The trigger source select is defined by the corresponding SRC_SEL bits of the module or task.

Trigger destination select

A module or task that process data needs to know that the following task in the chain is ready to receive the processed data. The user needs to specify what is the destination of the processed data. This is done by setting the corresponding DEST_SEL bits of the module or task.

Double buffering

The IPUv3EX supports double buffering in the system's memory. When a flow is processed frame by frame the first frame will be read from one location (BUF0) in the memory, the next frame will be read from another location in the memory (BUF1). The location in the memory of the buffers is defined by the EBA0 and EBA1 parameters of the corresponding channel in the CPMEM. The IDMAC use the correct buffer according to the DMA_CH_CUR_BUF_# signal. The FSU automatically toggles the DMA_CH_CUR_BUF_# to point on the correct buffer to be used by the channel. In order to work in double buffer mode the corresponding DMA_CH_DB_MODE_SEL_# needs to be set.

Alternative flow

Some of the IPUv3EX sub modules can handle 2 flows via them one is the main flow and the other is the alt flow. In order to support an alternate flow via the same module an alternative configuration should be used by the module. This includes

- Alternate registers including an alternative module's setup - this is handled by the SRM
- Alternate IDMAC settings: parameters in the CPMEM, separate alpha
- Alternate SRC_SEL as the source of the trigger may come from a different function.
- Alternate FSU settings (CUR_BUF, BUF_RDY, DB_MODE_SEL)
- Some of the display modules alternate settings are handled by programming an alternative set of registers.

The FSU handles the switch to the alternate flow, it controls the update of the alternate configuration and send the appropriate signals to other modules in the IPUv3EX indicating about a need to switch to the alternate configuration.

Once a frame is completed there is a chance that there are 2 buffers ready. One is the next buffer of the current flow and the other is the a buffer in an alternate flow. The FSU will automatically select between the next buffer to handle in a round-robin fashion.

IPU task chaining - Single flow

The diagram below illustrates task chaining in the IPUv3EX. In this example a single flow is handled

The Frames are coming from the same camera module. The frames are handled frame by Frame. The first frame arriving Frame0 is stored in BUF0 of the "INPUT BUFFER". Once the Frame0 is ready in BUF0, the processing module is triggered and data is read from BUF0 to the processing module. In the meantime, Frame1 coming from the camera is written to BUF1 of the "INPUT BUFFER". The processing module processes the data and stores the first frame on BUF0 of the "OUTPUT BUFFER". Once the processing is done and the frame is ready, the display module is triggered to pick the data and send it to the display.

The processing module will start working on the next frame once the data is ready in the “INPUT_BUFFER” and there’s a free buffer in the “OUTPUT_BUFFER”.

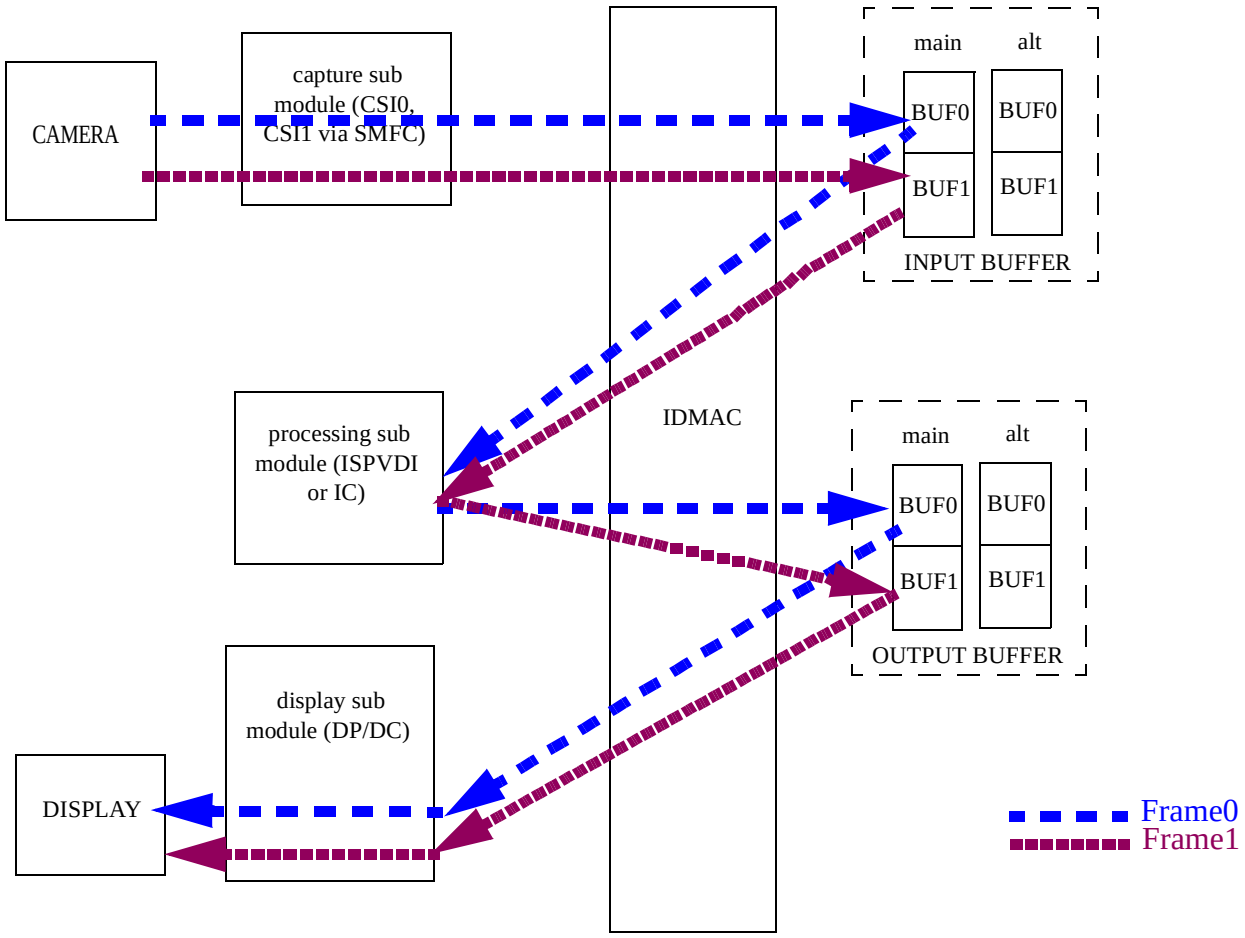


Figure 42-460. IPUv3EX **tasks’ chaining illustration - single flow**

IPUv3EX **task chaining - double flow**

The diagram below illustrates task chaining in the IPUv3EX. In this example double flow is handled. In addition to the flow described on the previous example, another flow is handled via the display modules. In that case the DC will handle 2 flows. Once sending Frame0 to the display is complete, the FSU will decide in whether to handle Frame1 or Frame0_ALT. The decision is made according the readiness of the other buffers in the memory, in case of 2 ready buffers (main flow and alternate) the FSU will switch between them in a round robin fashion.

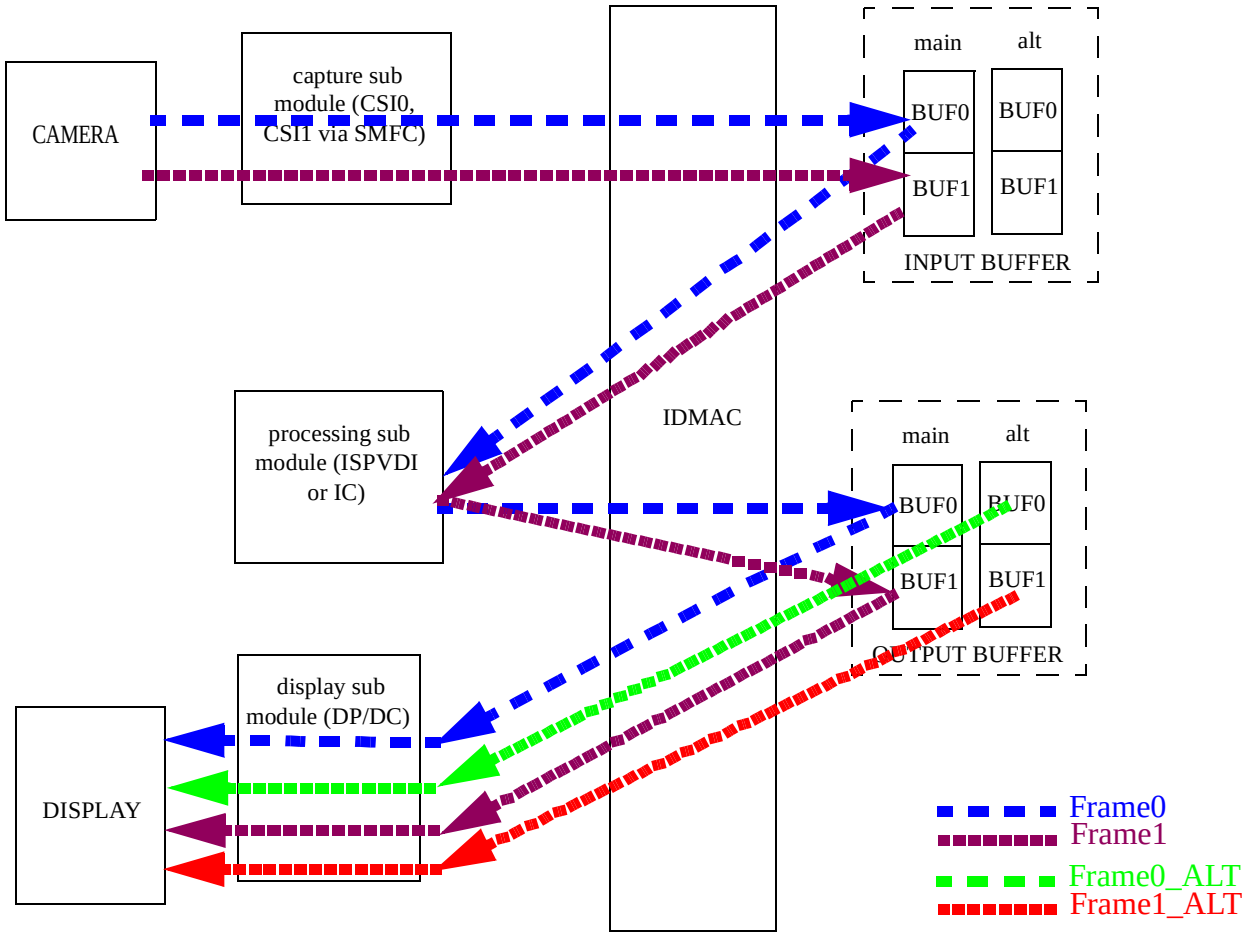


Figure 42-461. IPUv3EX **tasks' chaining illustration - double flow**

42.3.12.2.4 IPUv3EX **main flows**

IPUv3EX's flows can be partitioned into 5 groups

- CF - capture flows
- PF - processing flows
- SF - synchronous display flows
- AF - Asynchronous display flows
- DF - Direct flow from IC to the display

Tasks from different groups can be chained as illustrated below.

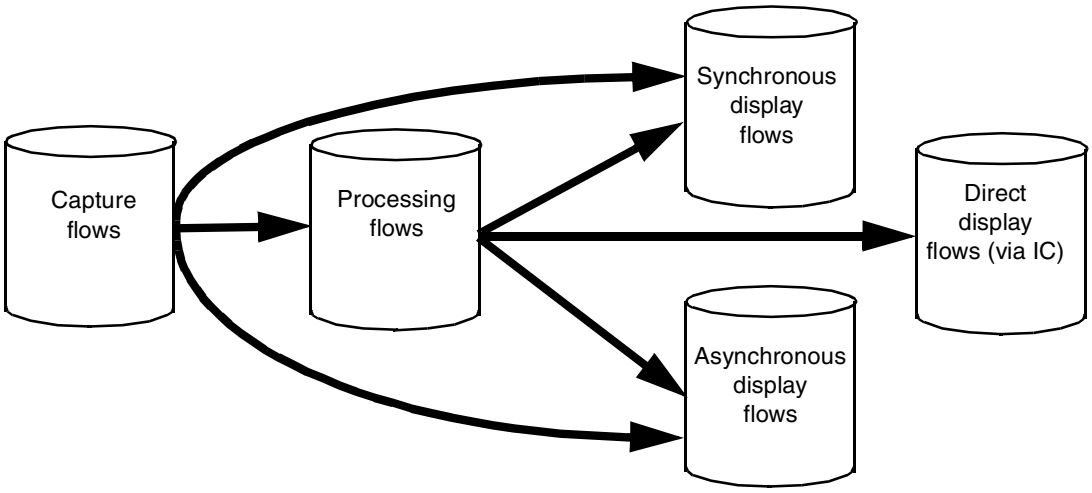


Figure 42-462. IPUv3EX **task flows chaining**

The tables below describe the most important use cases of chaining the IPUv3EX tasks. The tables show the main task for each group.

The physical DMA channel is the DMA channel that is used for the main flow - this is the IDMAC channel that is physically connected to the module - the CPMEM parameters should be configured according to the physical channel number. In case of an alternate flow then an alternate entry in the CPMEM should be configured as well

Table 42-428 describes capturing flows where data is captured from the sensor and sent to the memory without processing. This flows can be chained to the processing and display flows described on: Table 42-429, Table 42-430, Table 42-431 and on Table 42-432.

Table 42-428. IPUv3EX**'s capture flows**

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
1.	Capturing image from sensor and storing it in the memory (via SMFC) without processing	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3

Table 42-428. IPUv3EX's capture flows (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
2.	Capturing image from sensor and storing it in the memory (via IC) without processing	CSI0 or CSI1 --> MEM	---	---	IDMAC_CH_5
3.	Capturing interlaced input and storing it in the memory (via VDI) while performing video de-interlacing in the VDI.	CSI0 or CSI1 -> VDI --> MEM		IDMAC_CH_9 IDMAC_CH_10	IDMAC_CH_5 IDMAC_CH_13
		Interlaced input coming from one of the CSIs is sent to the memory without processing via channel #13. In addition 2 more inputs are read from the memory via channels 9 and 10. The processed image is written to the memory via ch 5 in progressive scan mode.			

Table 42-429 describe processing flows via the IC . This flows can start from the memory, can be chained to a capturing flow described on Table 42-428. The target of this flow can be chained to the display flows described on Table 42-430 and on Table 42-431.

Table 42-429. IPUv3EX's Processing flows

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
4.	Preprocessing image from sensor for encoding	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PRP ENC) --> MEM	IDMAC_CH_12	---	IDMAC_CH_20
5.	Preprocessing image from memory for encoding	MEM --> IC (PRP ENC) --> MEM	IDMAC_CH_12	---	IDMAC_CH_20
6.	Preprocessing image from sensor for encoding	CSI0 or CSI1 --> IC (PRP ENC) --> MEM	---	---	IDMAC_CH_20

Table 42-429. IPUv3EX's Processing flows (continued) (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
7.	Preprocessing + rotation of image from sensor for encoding	CSI0 or CSI1 -->SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PRP ENC) --> MEM	IDMAC_CH_12	---	IDMAC_CH_20
		MEM --> IC (ROT ENC) --> MEM	IDMAC_CH_45	---	IDMAC_CH_48
8.	Preprocessing + rotation of image from memory for encoding	MEM --> IC (PRP ENC) --> MEM	IDMAC_CH_12	---	IDMAC_CH_20
		MEM --> IC (ROT ENC) --> MEM	IDMAC_CH_45	---	IDMAC_CH_48
9.	Rotation and preprocessing of image from sensor for encoding	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (ROT ENC) --> MEM	IDMAC_CH_45	---	IDMAC_CH_48
		MEM --> IC (PRP ENC) --> MEM	IDMAC_CH_12	---	IDMAC_CH_20
10.	Preprocessing image from sensor for viewfinder	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PRP VF) --> MEM	IDMAC_CH_12	IDMAC_CH_14	IDMAC_CH_21
11.	Preprocessing image from memory for viewfinder	MEM --> IC (PRP VF) --> MEM	IDMAC_CH_12	IDMAC_CH_14	IDMAC_CH_21
12.	Preprocessing and rotation of image from sensor for viewfinder	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PRP VF) --> MEM	IDMAC_CH_12	IDMAC_CH_14	IDMAC_CH_21
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49

Table 42-429. IPUv3EX's Processing flows (continued) (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
13.	Rotation and preprocessing of image from sensor for viewfinder	CSI0 or CSI1 -->SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
		MEM --> IC (PRP VF) --> MEM	IDMAC_CH_12	IDMAC_CH_14	IDMAC_CH_21
14.	Preprocessing and rotation of image from sensor for viewfinder	MEM --> IC (PRP VF) --> MEM	IDMAC_CH_12	IDMAC_CH_14	IDMAC_CH_21
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
15.	Preprocessing image from sensor	CSI0 or CSI1 --> IC (PRP VF) --> MEM	---	IDMAC_CH_14	IDMAC_CH_21
16.	Preprocessing and rotation of image from sensor	CSI0 or CSI1 --> IC (PRP VF) --> MEM	---	IDMAC_CH_14	IDMAC_CH_21
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
17.	Postprocessing image	MEM --> IC (PP) --> MEM	IDMAC_CH_11	IDMAC_CH_15	IDMAC_CH_22
18.	Postprocessing and rotation of image	MEM --> IC (PP) --> MEM	IDMAC_CH_11	IDMAC_CH_14	IDMAC_CH_22
		MEM --> IC (ROT PP) -->MEM	IDMAC_CH_47	---	IDMAC_CH_50
19.	Rotation and postprocessing of image	MEM --> IC (ROT PP) -->MEM	IDMAC_CH_47	---	IDMAC_CH_50
		MEM --> IC (PP) --> MEM	IDMAC_CH_11	IDMAC_CH_14	IDMAC_CH_22
20.	Postprocessing image from sensor	CSI0 or CSI1 -->SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PP) --> MEM	IDMAC_CH_11	IDMAC_CH_15	IDMAC_CH_22

Table 42-429. IPUv3EX's Processing flows (continued) (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
21.	video de-interlacing in the VDI.	MEM -> VDI --> MEM	IDMAC_CH_8 IDMAC_CH_9 IDMAC_CH_10	--	IDMAC_CH_5
		Interlaced input coming from the memory via 3 channels 8, 9 and 10. The processed image is written to the memory via ch 5 in progressive scan mode.			
22.	video de-interlacing in the VDI. Then processing in the IC.	MEM -> VDI --> IC (PRP VF) --> MEM	IDMAC_CH_8 IDMAC_CH_9 IDMAC_CH_10	IDMAC_CH_14	IDMAC_CH_21
		Interlaced input coming from the memory via 3 channels 8, 9 and 10. The processed image is sent to the IC for further processing then it is sent to the memory via ch 21			
23.	video de-interlacing in the VDI, then preprocessing and rotation of image coming from memory	MEM --> VDI --> IC (PRP VF) --> MEM	IDMAC_CH_8 IDMAC_CH_9 IDMAC_CH_10	IDMAC_CH_14	IDMAC_CH_21
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
24.	video de-interlacing in the VDI, then preprocessing and rotation of image coming from CSI	CSI0 or CSI1 --> VDI--> IC (PRP VF) --> MEM	IDMAC_CH_9 IDMAC_CH_10	IDMAC_CH_14	IDMAC_CH_13 IDMAC_CH_21
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49

Table 42-430 describes the asynchronous display flows, this flows can be chained to the capture, processing and direct flows described on Table 42-428, Table 42-429 and on Table 42-432

Table 42-430. IPUv3EX synchronous display flows

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
1.	Synchronous display refresh via DC	MEM --> DC	IDMAC_CH_28	---	---
2.	Synchronous display refresh via DP	MEM --> DP	IDMAC_CH_23	---	---

Table 42-430. IPUv3EX synchronous display flows

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
3.	Synchronous display refresh via DC	MEM --> DC	IDMAC_CH_28	IDMAC_CH_44	---
		Comment: IDMAC_CH_44 is an optional mask Channel			
4.	Synchronous display refresh via DP	MEM --> DP	IDMAC_CH_23	IDMAC_CH_44	---
		Comment: IDMAC_CH_44 is an optional mask Channel			
5.	Synchronous display refresh via DP + combining in the DP	MEM --> DP SYNC(BG/FG)	IDMAC_CH_23	IDMAC_CH_27	---
		Comment: IDMAC_CH_23 is the main channel; IDMAC_CH_27 is used for combining of another plane.			
6.	Synchronous display refresh via DP	MEM --> DP SYNC(BG/FG)	IDMAC_CH_23 /IDMAC_CH_27	IDMAC_CH_44 (mask)	---
		Comment: IDMAC_CH_23 is the main channel; IDMAC_CH_27 is optional and can be used for combining of another plane. IDMAC_CH_44 is an optional mask Channel			

Table 42-431 describes the asynchronous display flows, this flows can be chained to the capture, processing and direct flows described on Table 42-428, Table 42-429 and on Table 42-432

Table 42-431. IPUv3EX Asynchronous display flows

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
1.	Asynchronous display refresh via DP	MEM --> DC	IDMAC_CH_24	---	---
2.	Asynchronous display refresh via DP	MEM --> DC	IDMAC_CH_24 /IDMAC_CH_29	---	---
		Comment: IDMAC_CH_29 is another input to the DP to be combined with data coming from IDMAC_CH_24			
3.	Asynchronous display refresh via DP in command buffer mode	MEM --> DC	IDMAC_CH_24 /IDMAC_CH_29	IDMAC_CH_42 (command)	---
		Comment: IDMAC_CH_29 is another input to the DP to be combined with data coming from IDMAC_CH_24			

Table 42-431. IPUv3EX Asynchronous display flows (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
4.	Asynchronous display refresh via DC	MEM --> DC	IDMAC_CH_28	---	---
5.	Asynchronous display refresh via DC	MEM --> DC	IDMAC_CH_41	---	---
6.	Asynchronous display refresh via DC	MEM --> DC	IDMAC_CH_28	IDMAC_CH_42 (command)	---
		Comment: IDMAC_CH_42 can be optionally used as command channel associated with IDMAC_CH_28			
7.	Asynchronous display refresh via DC	MEM --> DC	IDMAC_CH_41	IDMAC_CH_43 (command)	---
		Comment: IDMAC_CH_43 can be optionally used as command channel associated with IDMAC_CH_41			
8.	Asynchronous display refresh via DP	MEM --> DP ASYNC	IDMAC_CH_24	IDMAC_CH_42 (command)	---
		Comment: IDMAC_CH_42 can be optionally used as command channel associated with IDMAC_CH_24			
9.	Asynchronous display refresh via DP	MEM --> DC	IDMAC_CH_24	IDMAC_CH_43 (command)	---
		Comment: IDMAC_CH_43 can be optionally used as command channel associated with IDMAC_CH_24			
10.	Reading data from asynchronous display	DC --> MEM	---	---	IDMAC_CH_50

Table 42-432 describes direct flows via the IC. These are processing flows via the IC where the output is sent directly to the DMFC. From that point the any of the display flows described on Table 42-430 and on Table 42-431 can be chained.

Table 42-432. IIPUv3EX direct flows to the display via the IC

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
1.	Rotation and preprocessing of image from sensor for viewfinder and displaying it on synchronous display via DP	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
		MEM --> IC (PRP VF) --> DMFC	IDMAC_CH_12	IDMAC_CH_14	Direct to the DMFC
		MEM --> DP SYNC (BG/FG)	IDMAC_CH_23 /IDMAC_CH_27	IDMAC_CH_44 (mask)	---
2.	Rotation and preprocessing of image from sensor for viewfinder and displaying it on asynchronous display via DP	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (ROT VF) --> MEM	IDMAC_CH_46	---	IDMAC_CH_49
		MEM --> IC (PRP VF) --> DMFC	IDMAC_CH_12	IDMAC_CH_14	Direct to the DMFC
		MEM --> DP SYNC (BG/FG)	IDMAC_CH_24 /IDMAC_CH_29	command channel	---
3.	Preprocessing image from sensor for viewfinder and direct displaying it on asynchronous display	CSI0 or CSI1 --> SMFC --> MEM	---	---	IDMAC_CH_0 IDMAC_CH_1 IDMAC_CH_2 IDMAC_CH_3
		MEM --> IC (PRP VF) --> DC	IDMAC_CH_12	IDMAC_CH_14	---
4.	Preprocessing image from sensor for viewfinder and direct displaying it on asynchronous display	CSI0 or CSI1 --> IC (PRP VF) --> DC	---	IDMAC_CH_14	IDMAC_CH_21
5.	Preprocessing image from sensor for viewfinder and direct displaying it on asynchronous display via DP	CSI0 or CSI1 --> IC (PRP VF) --> DP ASYNC	---	IDMAC_CH_14	IDMAC_CH_21

Table 42-432. IIPUv3EX direct flows to the display via the IC (continued)

	Flow	Tasks chain	Physical DMA Channels		
			Video Input	Other Input	Output
6.	Capturing interlaced input via CSI, then perform video de-interlacing in the VDI. Then processing in the IC and direct displaying it on asynchronous display via DP	CSI0 or CSI1 -> VDI --> IC (PRP VF) --> DP ASYNC	--	IDMAC_CH_9 IDMAC_CH_10 IDMAC_CH_14	IDMAC_CH_13 IDMAC_CH_21
		Interlaced input coming from one of the CSIs is sent to the memory without processing via channel #13. In addition 2 more inputs are read from the memory via channels 9 and 10. The processed image is sent to the IC for further processing then it is sent to the display			
7.	Capturing interlaced input via CSI, then perform video de-interlacing in the VDI. Then processing in the IC and direct displaying it on asynchronous display	CSI0 or CSI1 -> VDI --> IC (PRP VF) --> DC	--	IDMAC_CH_9 IDMAC_CH_10 IDMAC_CH_14	IDMAC_CH_13 IDMAC_CH_21
		Interlaced input coming from one of the CSIs is sent to the memory without processing via channel #13. In addition 2 more inputs are read from the memory via channels 9 and 10. The processed image is sent to the IC for further processing then it is sent to the display			

42.3.12.2.5 Snooping

This function allows to reduce the rate of write accesses from the system memory to a smart display. When snooping is enabled, writing to the display is performed only if the corresponding system memory buffers has been changed. Any change of data in the system memory buffers is registered by the External Memory Controller of the chip. When writing to the system memory is performed, the External Memory Controller asserts the emi_snooping1 or emi_snooping2 pins of the IPUv3EX. After receiving these signals, the CM initiates transfer of the corresponding buffer to a display buffer, either in system's memory or in a smart display. Selecting the snooping mechanism as the trigger for the flow is done by setting the <>_SRC_SEL bits of the corresponding flow to snoop1 or snoop2 mode.

The emi_snooping1 and emi_snooping2 signals are synchronized to the IPUv3EX's clock. The synchronize on the emi_snooping2 signal can be bypassed by setting the bit SNOOP2_SYNC_BYP bit.

42.3.12.2.6 Automatic Window Refresh

By programming the <>_SRC_SEL bits of the corresponding flow to autoref mode, automatic refresh of a window on the smart display is enabled. This means that the flow is triggered any time the refresh counter completes its counting. The refresh period is defined via the AUTOREF_PER field (the time unit is 2^{17} periods of the HSP_CLK clock). The actual value of refresh period is equal to

$$T_{HSP}*2\text{^}17\ *(AUTO\_REF\_PER+1). \qquad \textit{Eqn. 42-15}$$

42.3.12.2.7 Autorefresh and snooping

Auto-refresh can be conditional. In this case, it is accomplished emi_snooping1 or emi_snooping2 pins of the IPUv3EX are asserted. This is defined by setting the <>_SRC_SEL bits to autoref+snoop1 or autoref_snoop2 mode.

42.3.12.2.8 Synchronization with A Video/Graphics source

The IPUv3EX supports direct synchronization with a video/graphics source (e.g. an integrated graphics accelerator, rendering graphics frames). Up to three frame buffers are supported for this synchronization. It is performed using the following signals

- The source instructs the IPUv3EX which frame should be transferred to the display by providing a 3 bit one-hot vector.
- The IPU notifies the source which frame is currently transferred to the display by providing a 3 bit one-hot vector.
- The IPU provides to the source an end-of-frame trigger (after which it switches to the frame indicated by the source).

This mechanism is provided for the main plane of the primary flow to the DP (ch 23). The SRC_SEL of this channel has to be defined as external source.

Triple buffer mode is enabled by setting the channel's corresponding bit on IPU_CH_TRB_MODE_SEL register. Setting this bit overrides the channel's corresponding settings on the IPU_CH_DB_MODE_SEL register. The CPMEM's settings of the corresponding channel can hold 2 pointers for 2 buffers (EBA0 and EBA1). The third pointer resides on an alternate entry in the CPMEM. The pointer to the alternate entry is defined by the channel's corresponding IDMAC_SUB_ADDR field. EBA0 of the alternate entry in the CPMEM is used as a third pointer. All the other settings of alternate entry in the CPMEM must be the same as the channel's standard entry in the CPMPM.

Once the channel reached end-of-frame event, the FSU automatically switch between pointers according to the signals coming from the source and notifies the source that the IPUv3EX reached the end-of-frame.

IPUv3EX has 2 identical bridges allowing direct synchronization with 2 sources. The 2 bridges support exactly the same protocol.

42.3.12.3 Interrupt Generator

The IG produces two interrupts to the MCU - the functional interrupt and the error interrupt. All the interrupts are maskable.

Table 42-433 describes the functional interrupts.

Table 42-433. Functional Interrupts Summary

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_1[0]	IDMAC	IDMAC_EOF_0	
IPU_INT_STAT_1[1]	IDMAC	IDMAC_EOF_1	
IPU_INT_STAT_1[2]	IDMAC	IDMAC_EOF_2	
IPU_INT_STAT_1[3]	IDMAC	IDMAC_EOF_3	
IPU_INT_STAT_1[4]	IDMAC	IDMAC_EOF_4	
IPU_INT_STAT_1[5]	IDMAC	IDMAC_EOF_5	
IPU_INT_STAT_1[6]	IDMAC	IDMAC_EOF_6	
IPU_INT_STAT_1[7]	IDMAC	IDMAC_EOF_7	
IPU_INT_STAT_1[8]	IDMAC	IDMAC_EOF_8	
IPU_INT_STAT_1[9]	IDMAC	IDMAC_EOF_9	
IPU_INT_STAT_1[10]	IDMAC	IDMAC_EOF_10	
IPU_INT_STAT_1[11]	IDMAC	IDMAC_EOF_11	
IPU_INT_STAT_1[12]	IDMAC	IDMAC_EOF_12	
IPU_INT_STAT_1[13]	IDMAC	IDMAC_EOF_13	
IPU_INT_STAT_1[14]	IDMAC	IDMAC_EOF_14	
IPU_INT_STAT_1[15]	IDMAC	IDMAC_EOF_15	
IPU_INT_STAT_1[17]	IDMAC	IDMAC_EOF_17	
IPU_INT_STAT_1[18]	IDMAC	IDMAC_EOF_18	
IPU_INT_STAT_1[20]	IDMAC	IDMAC_EOF_20	
IPU_INT_STAT_1[21]	IDMAC	IDMAC_EOF_21	
IPU_INT_STAT_1[22]	IDMAC	IDMAC_EOF_22	
IPU_INT_STAT_1[23]	IDMAC	IDMAC_EOF_23	
IPU_INT_STAT_1[24]	IDMAC	IDMAC_EOF_24	
IPU_INT_STAT_1[27]	IDMAC	IDMAC_EOF_27	
IPU_INT_STAT_1[28]	IDMAC	IDMAC_EOF_28	
IPU_INT_STAT_1[29]	IDMAC	IDMAC_EOF_29	
IPU_INT_STAT_1[31]	IDMAC	IDMAC_EOF_31	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_2[1]	IDMAC	IDMAC_EOF_33	
IPU_INT_STAT_2[8]	IDMAC	IDMAC_EOF_40	
IPU_INT_STAT_2[9]	IDMAC	IDMAC_EOF_41	
IPU_INT_STAT_2[10]	IDMAC	IDMAC_EOF_42	
IPU_INT_STAT_2[11]	IDMAC	IDMAC_EOF_43	
IPU_INT_STAT_2[12]	IDMAC	IDMAC_EOF_44	
IPU_INT_STAT_2[13]	IDMAC	IDMAC_EOF_45	
IPU_INT_STAT_2[14]	IDMAC	IDMAC_EOF_46	
IPU_INT_STAT_2[15]	IDMAC	IDMAC_EOF_47	
IPU_INT_STAT_2[16]	IDMAC	IDMAC_EOF_48	
IPU_INT_STAT_2[17]	IDMAC	IDMAC_EOF_49	
IPU_INT_STAT_2[18]	IDMAC	IDMAC_EOF_50	
IPU_INT_STAT_2[19]	IDMAC	IDMAC_EOF_51	
IPU_INT_STAT_2[20]	IDMAC	IDMAC_EOF_52	
IPU_INT_STAT_3[0]	IDMAC	IDMAC_NFACK_0	
IPU_INT_STAT_3[1]	IDMAC	IDMAC_NFACK_1	
IPU_INT_STAT_3[2]	IDMAC	IDMAC_NFACK_2	
IPU_INT_STAT_3[3]	IDMAC	IDMAC_NFACK_3	
IPU_INT_STAT_3[4]	IDMAC	IDMAC_NFACK_4	
IPU_INT_STAT_3[5]	IDMAC	IDMAC_NFACK_5	
IPU_INT_STAT_3[6]	IDMAC	IDMAC_NFACK_6	
IPU_INT_STAT_3[7]	IDMAC	IDMAC_NFACK_7	
IPU_INT_STAT_3[8]	IDMAC	IDMAC_NFACK_8	
IPU_INT_STAT_3[9]	IDMAC	IDMAC_NFACK_9	
IPU_INT_STAT_3[10]	IDMAC	IDMAC_NFACK_10	
IPU_INT_STAT_3[11]	IDMAC	IDMAC_NFACK_11	
IPU_INT_STAT_3[12]	IDMAC	IDMAC_NFACK_12	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_3[13]	IDMAC	IDMAC_NFACK_13	
IPU_INT_STAT_3[14]	IDMAC	IDMAC_NFACK_14	
IPU_INT_STAT_3[15]	IDMAC	IDMAC_NFACK_15	
IPU_INT_STAT_3[17]	IDMAC	IDMAC_NFACK_17	
IPU_INT_STAT_3[18]	IDMAC	IDMAC_NFACK_18	
IPU_INT_STAT_3[20]	IDMAC	IDMAC_NFACK_20	
IPU_INT_STAT_3[21]	IDMAC	IDMAC_NFACK_21	
IPU_INT_STAT_3[22]	IDMAC	IDMAC_NFACK_22	
IPU_INT_STAT_3[23]	IDMAC	IDMAC_NFACK_23	
IPU_INT_STAT_3[24]	IDMAC	IDMAC_NFACK_24	
IPU_INT_STAT_3[27]	IDMAC	IDMAC_NFACK_27	
IPU_INT_STAT_3[28]	IDMAC	IDMAC_NFACK_28	
IPU_INT_STAT_3[29]	IDMAC	IDMAC_NFACK_29	
IPU_INT_STAT_3[31]	IDMAC	IDMAC_NFACK_31	
IPU_INT_STAT_4[1]	IDMAC	IDMAC_NFACK_33	
IPU_INT_STAT_4[8]	IDMAC	IDMAC_NFACK_40	
IPU_INT_STAT_4[9]	IDMAC	IDMAC_NFACK_41	
IPU_INT_STAT_4[10]	IDMAC	IDMAC_NFACK_42	
IPU_INT_STAT_4[11]	IDMAC	IDMAC_NFACK_43	
IPU_INT_STAT_4[12]	IDMAC	IDMAC_NFACK_44	
IPU_INT_STAT_4[13]	IDMAC	IDMAC_NFACK_45	
IPU_INT_STAT_4[14]	IDMAC	IDMAC_NFACK_46	
IPU_INT_STAT_4[15]	IDMAC	IDMAC_NFACK_47	
IPU_INT_STAT_4[16]	IDMAC	IDMAC_NFACK_48	
IPU_INT_STAT_4[17]	IDMAC	IDMAC_NFACK_49	
IPU_INT_STAT_4[18]	IDMAC	IDMAC_NFACK_50	
IPU_INT_STAT_4[19]	IDMAC	IDMAC_NFACK_51	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_4[20]	IDMAC	IDMAC_NFACK_52	
IPU_INT_STAT_7[23]	IDMAC	IDMAC_EOS_23	
IPU_INT_STAT_7[24]	IDMAC	IDMAC_EOS_24	
IPU_INT_STAT_7[27]	IDMAC	IDMAC_EOS_27	
IPU_INT_STAT_7[28]	IDMAC	IDMAC_EOS_28	
IPU_INT_STAT_7[29]	IDMAC	IDMAC_EOS_29	
IPU_INT_STAT_7[31]	IDMAC	IDMAC_EOS_31	
IPU_INT_STAT_8[2]	IDMAC	IDMAC_EOS_33	
IPU_INT_STAT_8[9]	IDMAC	IDMAC_EOS_41	
IPU_INT_STAT_8[10]	IDMAC	IDMAC_EOS_42	
IPU_INT_STAT_8[11]	IDMAC	IDMAC_EOS_43	
IPU_INT_STAT_8[12]	IDMAC	IDMAC_EOS_44	
IPU_INT_STAT_8[19]	IDMAC	IDMAC_EOS_51	
IPU_INT_STAT_8[20]	IDMAC	IDMAC_EOS_52	
IPU_INT_STAT_11[0]	IDMAC	IDMAC_EOBND_0	
IPU_INT_STAT_11[1]	IDMAC	IDMAC_EOBND_1	
IPU_INT_STAT_11[2]	IDMAC	IDMAC_EOBND_2	
IPU_INT_STAT_11[3]	IDMAC	IDMAC_EOBND_3	
IPU_INT_STAT_11[5]	IDMAC	IDMAC_EOBND_5	
IPU_INT_STAT_11[11]	IDMAC	IDMAC_EOBND_11	
IPU_INT_STAT_11[12]	IDMAC	IDMAC_EOBND_12	
IPU_INT_STAT_11[20]	IDMAC	IDMAC_EOBND_20	
IPU_INT_STAT_11[21]	IDMAC	IDMAC_EOBND_21	
IPU_INT_STAT_11[22]	IDMAC	IDMAC_EOBND_22	
IPU_INT_STAT_12[13]	IDMAC	IDMAC_EOBND_45	
IPU_INT_STAT_12[14]	IDMAC	IDMAC_EOBND_46	
IPU_INT_STAT_12[15]	IDMAC	IDMAC_EOBND_47	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_12[16]	IDMAC	IDMAC_EOBND_48	
IPU_INT_STAT_12[17]	IDMAC	IDMAC_EOBND_49	
IPU_INT_STAT_12[18]	IDMAC	IDMAC_EOBND_50	
IPU_INT_STAT_13[0]	IDMAC	IDMAC_TH_0	
IPU_INT_STAT_13[1]	IDMAC	IDMAC_TH_1	
IPU_INT_STAT_13[2]	IDMAC	IDMAC_TH_2	
IPU_INT_STAT_13[3]	IDMAC	IDMAC_TH_3	
IPU_INT_STAT_13[4]	IDMAC	IDMAC_TH_4	
IPU_INT_STAT_13[5]	IDMAC	IDMAC_TH_5	
IPU_INT_STAT_13[8]	IDMAC	IDMAC_TH_8	
IPU_INT_STAT_13[9]	IDMAC	IDMAC_TH_9	
IPU_INT_STAT_13[10]	IDMAC	IDMAC_TH_10	
IPU_INT_STAT_13[11]	IDMAC	IDMAC_TH_11	
IPU_INT_STAT_13[12]	IDMAC	IDMAC_TH_12	
IPU_INT_STAT_13[13]	IDMAC	IDMAC_TH_13	
IPU_INT_STAT_13[14]	IDMAC	IDMAC_TH_14	
IPU_INT_STAT_13[15]	IDMAC	IDMAC_TH_15	
IPU_INT_STAT_13[17]	IDMAC	IDMAC_TH_17	
IPU_INT_STAT_13[18]	IDMAC	IDMAC_TH_18	
IPU_INT_STAT_13[20]	IDMAC	IDMAC_TH_20	
IPU_INT_STAT_13[21]	IDMAC	IDMAC_TH_21	
IPU_INT_STAT_13[22]	IDMAC	IDMAC_TH_22	
IPU_INT_STAT_13[23]	IDMAC	IDMAC_TH_23	
IPU_INT_STAT_13[24]	IDMAC	IDMAC_TH_24	
IPU_INT_STAT_13[27]	IDMAC	IDMAC_TH_27	
IPU_INT_STAT_13[28]	IDMAC	IDMAC_TH_28	
IPU_INT_STAT_13[29]	IDMAC	IDMAC_TH_29	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_13[31]	IDMAC	IDMAC_TH_31	
IPU_INT_STAT_14[1]	IDMAC	IDMAC_TH_33	
IPU_INT_STAT_14[8]	IDMAC	IDMAC_TH_40	
IPU_INT_STAT_14[9]	IDMAC	IDMAC_TH_41	
IPU_INT_STAT_14[10]	IDMAC	IDMAC_TH_42	
IPU_INT_STAT_14[11]	IDMAC	IDMAC_TH_43	
IPU_INT_STAT_14[12]	IDMAC	IDMAC_TH_44	
IPU_INT_STAT_14[13]	IDMAC	IDMAC_TH_45	
IPU_INT_STAT_14[14]	IDMAC	IDMAC_TH_46	
IPU_INT_STAT_14[15]	IDMAC	IDMAC_TH_47	
IPU_INT_STAT_14[16]	IDMAC	IDMAC_TH_48	
IPU_INT_STAT_14[17]	IDMAC	IDMAC_TH_49	
IPU_INT_STAT_14[18]	IDMAC	IDMAC_TH_50	
IPU_INT_STAT_14[19]	IDMAC	IDMAC_TH_51	
IPU_INT_STAT_14[20]	IDMAC	IDMAC_TH_52	
IPU_INT_STAT_15[0]	CM	IPU_SNOOPING1_INT	
IPU_INT_STAT_15[1]	CM	IPU_SNOOPING2_INT	
IPU_INT_STAT_15[2]	DP	DP_SF_START	
IPU_INT_STAT_15[3]	DP	DP_SF_END	
IPU_INT_STAT_15[4]	DP	DP_ASF_START	
IPU_INT_STAT_15[5]	DP	DP_ASF_END	
IPU_INT_STAT_15[6]	DP	DP_SF_BRAKE	
IPU_INT_STAT_15[7]	DP	DP_ASF_BRAKE	
IPU_INT_STAT_15[8]	DC	DC_FC_0	
IPU_INT_STAT_15[9]	DC	DC_FC_1	
IPU_INT_STAT_15[10]	DC	DC_FC_2	
IPU_INT_STAT_15[11]	DC	DC_FC_3	
IPU_INT_STAT_15[12]	DC	DC_FC_4	
IPU_INT_STAT_15[13]	DC	DC_FC_6	

Table 42-433. Functional Interrupts Summary (continued)

Location	Sub-module	Interrupt status bit name	Description
IPU_INT_STAT_15[14]	DC	DI_VSYNC_PRE_0	
IPU_INT_STAT_15[15]	DC	DI_VSYNC_PRE_1	
IPU_INT_STAT_15[16]	DC	DC_DP_START	
IPU_INT_STAT_15[17]	DC	DC_ASYNC_STOP	
IPU_INT_STAT_15[18]	DI0	DI0_DISP_CLK_EN_PRE	
IPU_INT_STAT_15[19]	DI0	DI0_CNT_EN_PRE_1	
IPU_INT_STAT_15[20]	DI0	DI0_CNT_EN_PRE_2	
IPU_INT_STAT_15[21]	DI0	DI0_CNT_EN_PRE_3	
IPU_INT_STAT_15[22]	DI0	DI0_CNT_EN_PRE_4	
IPU_INT_STAT_15[23]	DI0	DI0_CNT_EN_PRE_5	
IPU_INT_STAT_15[24]	DI0	DI0_CNT_EN_PRE_6	
IPU_INT_STAT_15[25]	DI0	DI0_CNT_EN_PRE_7	
IPU_INT_STAT_15[26]	DI0	DI0_CNT_EN_PRE_8	
IPU_INT_STAT_15[27]	DI0	DI0_CNT_EN_PRE_9	
IPU_INT_STAT_15[28]	DI0	DI0_CNT_EN_PRE_10	
IPU_INT_STAT_15[29]	DI1	DI1_DISP_CLK_EN_PRE	
IPU_INT_STAT_15[30]	DI1	DI1_CNT_EN_PRE_3	
IPU_INT_STAT_15[31]	DI1	DI1_CNT_EN_PRE_8	

Table 42-434 describes the error interrupts. The panic column indicates if this signal is part of the logic generating the ipu_panic signal. The ipu_panic signal can be used for indicating about errors that are result of data rate problems. Such problems may be a result of the IPUv3EX running in slower clock then required by the use case. This signal can be used in order to indicate the system that the IPUv3EX can't handle the desired data rate. In that case the system may need to increase the clock to the IPUv3EX or simplify the use case.

Table 42-434. Error Interrupts Summary

Location	Sub-module	Interrupt Status name	panic	Description
IPU_INT_STAT_5[0]	IDMAC	IDMAC_NFB4EOF_ERR_0	YES	
IPU_INT_STAT_5[1]	IDMAC	IDMAC_NFB4EOF_ERR_1	YES	

Table 42-434. Error Interrupts Summary (continued)

Location	Sub-module	Interrupt Status name	panic	Description
IPU_INT_STAT_5[2]	IDMAC	IDMAC_NFB4EOF_ERR_2	YES	
IPU_INT_STAT_5[3]	IDMAC	IDMAC_NFB4EOF_ERR_3	YES	
IPU_INT_STAT_5[4]	IDMAC	IDMAC_NFB4EOF_ERR_4	YES	
IPU_INT_STAT_5[5]	IDMAC	IDMAC_NFB4EOF_ERR_5	YES	
IPU_INT_STAT_5[6]	IDMAC	IDMAC_NFB4EOF_ERR_6	YES	
IPU_INT_STAT_5[7]	IDMAC	IDMAC_NFB4EOF_ERR_7	YES	
IPU_INT_STAT_5[8]	IDMAC	IDMAC_NFB4EOF_ERR_8	YES	
IPU_INT_STAT_5[9]	IDMAC	IDMAC_NFB4EOF_ERR_9	YES	
IPU_INT_STAT_5[10]	IDMAC	IDMAC_NFB4EOF_ERR_10	YES	
IPU_INT_STAT_5[11]	IDMAC	IDMAC_NFB4EOF_ERR_11	YES	
IPU_INT_STAT_5[12]	IDMAC	IDMAC_NFB4EOF_ERR_12	YES	
IPU_INT_STAT_5[13]	IDMAC	IDMAC_NFB4EOF_ERR_13	YES	
IPU_INT_STAT_5[14]	IDMAC	IDMAC_NFB4EOF_ERR_14	YES	
IPU_INT_STAT_5[15]	IDMAC	IDMAC_NFB4EOF_ERR_15	YES	
IPU_INT_STAT_5[17]	IDMAC	IDMAC_NFB4EOF_ERR_17	YES	
IPU_INT_STAT_5[18]	IDMAC	IDMAC_NFB4EOF_ERR_18	YES	
IPU_INT_STAT_5[20]	IDMAC	IDMAC_NFB4EOF_ERR_20	YES	
IPU_INT_STAT_5[21]	IDMAC	IDMAC_NFB4EOF_ERR_21	YES	
IPU_INT_STAT_5[22]	IDMAC	IDMAC_NFB4EOF_ERR_22	YES	
IPU_INT_STAT_5[23]	IDMAC	IDMAC_NFB4EOF_ERR_23	YES	
IPU_INT_STAT_5[24]	IDMAC	IDMAC_NFB4EOF_ERR_24	YES	
IPU_INT_STAT_5[27]	IDMAC	IDMAC_NFB4EOF_ERR_27	YES	
IPU_INT_STAT_5[28]	IDMAC	IDMAC_NFB4EOF_ERR_28	YES	
IPU_INT_STAT_5[29]	IDMAC	IDMAC_NFB4EOF_ERR_29	YES	
IPU_INT_STAT_5[31]	IDMAC	IDMAC_NFB4EOF_ERR_31	YES	
IPU_INT_STAT_6[1]	IDMAC	IDMAC_NFB4EOF_ERR_33	YES	
IPU_INT_STAT_6[8]	IDMAC	IDMAC_NFB4EOF_ERR_40	YES	
IPU_INT_STAT_6[9]	IDMAC	IDMAC_NFB4EOF_ERR_41	YES	
IPU_INT_STAT_6[10]	IDMAC	IDMAC_NFB4EOF_ERR_42	YES	
IPU_INT_STAT_6[11]	IDMAC	IDMAC_NFB4EOF_ERR_43	YES	
IPU_INT_STAT_6[12]	IDMAC	IDMAC_NFB4EOF_ERR_44	YES	
IPU_INT_STAT_6[13]	IDMAC	IDMAC_NFB4EOF_ERR_45	YES	

Table 42-434. Error Interrupts Summary (continued)

Location	Sub-module	Interrupt Status name	panic	Description
IPU_INT_STAT_6[14]	IDMAC	IDMAC_NFB4EOF_ERR_46	YES	
IPU_INT_STAT_6[15]	IDMAC	IDMAC_NFB4EOF_ERR_47	YES	
IPU_INT_STAT_6[16]	IDMAC	IDMAC_NFB4EOF_ERR_48	YES	
IPU_INT_STAT_6[17]	IDMAC	IDMAC_NFB4EOF_ERR_49	YES	
IPU_INT_STAT_6[18]	IDMAC	IDMAC_NFB4EOF_ERR_50	YES	
IPU_INT_STAT_6[19]	IDMAC	IDMAC_NFB4EOF_ERR_51	YES	
IPU_INT_STAT_6[20]	IDMAC	IDMAC_NFB4EOF_ERR_52	YES	
IPU_INT_STAT_9[0]	IC	VDI_FIFO1_OVF	YES	
IPU_INT_STAT_9[26]	IC	IC_BAYER_BUF_OVF	YES	
IPU_INT_STAT_9[27]	IC	IC_ENC_BUF_OVF	YES	
IPU_INT_STAT_9[28]	IC	IC_VF_BUF_OVF	YES	
IPU_INT_STAT_9[30]	CSI0	CSI0_PUPE	YES	
IPU_INT_STAT_9[31]	CSI0	CSI1_PUPE	YES	
IPU_INT_STAT_10[0]	SMFC	SMFC0_FRM_LOST	YES	
IPU_INT_STAT_10[1]	SMFC	SMFC1_FRM_LOST	YES	
IPU_INT_STAT_10[2]	SMFC	SMFC2_FRM_LOST	YES	
IPU_INT_STAT_10[3]	SMFC	SMFC3_FRM_LOST	YES	
IPU_INT_STAT_10[16]	DC	DC_TEARING_ERR_1	YES	
IPU_INT_STAT_10[17]	DC	DC_TEARING_ERR_2	YES	
IPU_INT_STAT_10[18]	DC	DC_TEARING_ERR_6	YES	
IPU_INT_STAT_10[19]	DI0	DI0_SYNC_DISP_ERR	YES	
IPU_INT_STAT_10[20]	DI1	DI1_SYNC_DISP_ERR	YES	
IPU_INT_STAT_10[21]	DI0	DI0_TIME_OUT_ERR	YES	
IPU_INT_STAT_10[22]	DI1	DI1_TIME_OUT_ERR	YES	
IPU_INT_STAT_10[24]	IC	IC_VF_FRM_LOST_ERR	YES	
IPU_INT_STAT_10[25]	IC	IC_ENC_FRM_LOST_ERR	YES	
IPU_INT_STAT_10[26]	IC	IC_BAYER_FRM_LOST_ERR	YES	
IPU_INT_STAT_10[28]	CM	NON_PRIVILEGED_ACC_ERR	NO	
IPU_INT_STAT_10[29]	IDMAC	AXIW_ERR	NO	
IPU_INT_STAT_10[30]	IDMAC	AXIR_ERR	NO	

42.3.12.4 SDMA event generator

The IPUv3EX provides an SDMA event signal that can be used as trigger to the SoC's SDMA. IPUv3EX routes an internal event to the SDMA event signal. The internal event causing the assertion of the SDMA signal is enabled by setting the corresponding bit on the IPU_SDMA_EVENT_# registers. The user is allowed to enable multiple events. When one of these events occurs the ipu_sdma_event signal will be asserted. Similar to interrupts, the ipu_sdma_event signal is cleared by writing one to the corresponding bit in the IPU_INT_STAT_# registers.

It is not recommended to use the same internal event for a simultaneous generation of an interrupt signal and an SDMA event. This will require special software care when clearing the corresponding bit in the IPU_INT_STAT_# registers.

42.3.12.5 General Configuration Registers

The GCR contains a set of control/status/data registers. It provides IPUv3EX interface to the AHB slave bus. The HCLK rate is equal to the HSP_CLK rate. The detail description of the registers is found in Section 42.2.1, "Memory Map"."

42.3.12.6 Shadow Registers Module (SRM)

IPUv3EX supports frame by frame task switching. This means that a sub module can handle a frame with one configuration and handle the following frame with different configuration. Changing the configuration is done by updating the module's parameters. In order to allow automatic flow without a need of the SW to update all the parameters at the frame boundaries.A Register of a module that has the shadowing capabilities has a shadow register file that resides in the Shadow Register Memory.

The modules supporting this function may use it in one of the following ways:

42.3.12.6.1 Switching between 2 flows

Upon request from the FSU the SRM switches between the registers and the content in the Shadow Register Memory. When a module use one of the configuration SW is allowed to update the parameters in the memory. This is normally used when 2 flows are supported via one module.

- The current flow's configuration is stored in the module's registers
- The alternate flow configuration is stored in the SRM.
- When switching between flows the current flow's configuration is stored in the SRM. The alternate flow's configuration is written to the module's registers. When the alternate flow ends the configurations are swapped again.

This process is fully controlled by the FSU

42.3.12.6.2 Updating parameters between frames

This mode is used when the user needs to update the parameters of the current task being processed by the module. The updates can't affect the frame that is currently being processed. The update is effective only on the next frame. The SRM performs the parameters update only when there's a specific request by the user.

- The current frame's configuration is stored in the module's registers
- The next frame's configuration is stored in the SRM.
- On frame's boundary the SRM reads the new configuration from the SRM and writes it to the module's registers. The old configuration is lost.

42.3.12.6.3 Updating the memory

In order to avoid data coherency problems, the user should set a flag indicating that he is currently updating the memory region of a specific module. When the flag is set the SRM will not attempt to replace the parameters relevant to the specific module currently being updated by the user.

The user should clear this flag when he completes the update, and the parameters are ready to be used.

The flag is the corresponding SRM_MODE field for each module. This field controls the SRM logic that handles the modules registers

- 00 - MCU is allowed to access the module's region in the RAM; The automatic swapping mechanism is disabled.
- 01 - The SRM logic is controlled by the FSU. The update will be done of the next frame.
- 10 - The SRM logic is controlled by the FSU. Registers are swapped continuously frame by frame
- 11 - Update now. The SRM is controlled by the MCU. The Register will be update now

Each module uses the SRM mechanism according to the modules behavior, the table below summarizes the SRM support for each module

Table 42-435. SRM support per module

Module	Switching between 2 flows support	Updating parameters between frames	comment
CSI1, CSI0	NO	YES	SRM_MODE can be 00 or 01. The update will happen only once. When set to 01: after the update the state machine is automatically moved to 00 mode. It is recommended to make sure that the first frame has started by polling the corresponding NFACK bit before setting the SRM_MODE
DI1, DI0	NO	YES	DI0, DI1 parameters are needed when clock change is performed (See: 42.3.12.9 Clock Change procedure). SRM_MODE can be 00 or 01. The update will happen only once. When set to 01: after the update the state machine is automatically moved to 00 mode

Table 42-435. SRM support per module

Module	Switching between 2 flows support	Updating parameters between frames	comment
DC	YES	NO	SRM_MODE can be 00 or 10.
DP	YES	NO	SRM_MODE can be 00, 10 or 11. When set to 11: after the update the state machine is automatically moved to 10 mode 10 is not supported for SYNC flows

In order to update parameters the user should monitor the SRM_BUSY bit of the corresponding module. When the SRM is not busy the user should set the SRM_MODE to 00. The user will now update the register file in the memory. When done the user should switch the SRM_MODE field to the desired mode.

42.3.12.6.4 SRM priority

The SRM updates the registers according to a pre defined priority. The priority is set according to the SRM_PRI bits of each module. The user must set a unique value for each module.

42.3.12.6.5 SRM entries mapping

The table below maps any IPUv3EX register to an address in the SRM. The registers marked as NONE do not have an SRM entry

PG column indicates if this register is saved during power gating mode

LPSR column indicates if this register is swapped during low power screen refresh mode (LPSR)

.

Table 42-436. IPUv3EX SRM entries mapping

Register's name	Register's address	SRM entry	PG	LPSR
IPU_CONF	0x1E000000	NONE	YES	NO
IPU_SISG_CTRL0	0x1E000004	NONE	YES	NO
IPU_SISG_CTRL1	0x1E000008	NONE	YES	NO
IPU_SISG_SET_1	0x1E00000C	NONE	YES	NO
IPU_SISG_SET_2	0x1E000010	NONE	YES	NO
IPU_SISG_SET_3	0x1E000014	NONE	YES	NO
IPU_SISG_SET_4	0x1E000018	NONE	YES	NO
IPU_SISG_SET_5	0x1E00001C	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
IPU_SISG_SET_6	0x1E000020	NONE	YES	NO
IPU_SISG_CLR_1	0x1E000024	NONE	YES	NO
IPU_SISG_CLR_2	0x1E000028	NONE	YES	NO
IPU_SISG_CLR_3	0x1E00002C	NONE	YES	NO
IPU_SISG_CLR_4	0x1E000030	NONE	YES	NO
IPU_SISG_CLR_5	0x1E000034	NONE	YES	NO
IPU_SISG_CLR_6	0x1E000038	NONE	YES	NO
IPU_INT_CTRL_1	0x1E00003C	NONE	YES	NO
IPU_INT_CTRL_2	0x1E000040	NONE	YES	NO
IPU_INT_CTRL_3	0x1E000044	NONE	YES	NO
IPU_INT_CTRL_4	0x1E000048	NONE	YES	NO
IPU_INT_CTRL_5	0x1E00004C	NONE	YES	NO
IPU_INT_CTRL_6	0x1E000050	NONE	YES	NO
IPU_INT_CTRL_7	0x1E000054	NONE	YES	NO
IPU_INT_CTRL_8	0x1E000058	NONE	YES	NO
IPU_INT_CTRL_9	0x1E00005C	NONE	YES	NO
IPU_INT_CTRL_10	0x1E000060	NONE	YES	NO
IPU_INT_CTRL_11	0x1E000064	NONE	YES	NO
IPU_INT_CTRL_12	0x1E000068	NONE	YES	NO
IPU_INT_CTRL_13	0x1E00006C	NONE	YES	NO
IPU_INT_CTRL_14	0x1E000070	NONE	YES	NO
IPU_INT_CTRL_15	0x1E000074	NONE	YES	NO
IPU_SDMA_EVENT_1	0x1E000078	NONE	YES	NO
IPU_SDMA_EVENT_2	0x1E00007C	NONE	YES	NO
IPU_SDMA_EVENT_3	0x1E000080	NONE	YES	NO
IPU_SDMA_EVENT_4	0x1E000084	NONE	YES	NO
IPU_SDMA_EVENT_7	0x1E000088	NONE	YES	NO
IPU_SDMA_EVENT_8	0x1E00008C	NONE	YES	NO
IPU_SDMA_EVENT_11	0x1E000090	NONE	YES	NO
IPU_SDMA_EVENT_12	0x1E000094	NONE	YES	NO
IPU_SDMA_EVENT_13	0x1E000098	NONE	YES	NO
IPU_SDMA_EVENT_14	0x1E00009C	NONE	YES	NO
IPU_SRM_PRI1	0x1E0000A0	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
IPU_SRM_PRI2	0x1E0000A4	NONE	YES	NO
IPU_FS_PROC_FLOW1	0x1E0000A8	NONE	YES	NO
IPU_FS_PROC_FLOW2	0x1E0000AC	NONE	YES	NO
IPU_FS_PROC_FLOW3	0x1E0000B0	NONE	YES	NO
IPU_FS_DISP_FLOW1	0x1E0000B4	NONE	YES	NO
IPU_FS_DISP_FLOW2	0x1E0000B8	NONE	YES	NO
IPU_SKIP	0x1E0000BC	NONE	YES	NO
IPU_DISP_ALT_CONF	0x1E0000C0	NONE	YES	NO
IPU_DISP_GEN	0x1E0000C4	NONE	YES	NO
IPU_DISP_ALT1	0x1E0000C8	NONE	YES	NO
IPU_DISP_ALT2	0x1E0000CC	NONE	YES	NO
IPU_DISP_ALT3	0x1E0000D0	NONE	YES	NO
IPU_DISP_ALT4	0x1E0000D4	NONE	YES	NO
IPU_SNOOP	0x1E0000D8	NONE	YES	NO
IPU_MEM_RST	0x1E0000DC	NONE	YES	NO
IPU_PM	0x1E0000E0	NONE	YES	NO
IPU_GPR	0x1E0000E4	NONE	YES	NO
IPU_CH_DB_MODE_SEL_0	0x1E000150	NONE	YES	NO
IPU_CH_DB_MODE_SEL_1	0x1E000154	NONE	YES	NO
IPU_ALT_CH_DB_MODE_SEL_0	0x1E000168	NONE	YES	NO
IPU_ALT_CH_DB_MODE_SEL_1	0x1E00016C	NONE	YES	NO
IPU_CH_TRB_MODE_SEL_0	0x1E000178	NONE	YES	NO
IPU_CH_TRB_MODE_SEL_1	0x1E00017C	NONE	YES	NO
IPU_INT_STAT_1	0x1E000200	NONE	NO	NO
IPU_INT_STAT_2	0x1E000204	NONE	NO	NO
IPU_INT_STAT_3	0x1E000208	NONE	NO	NO
IPU_INT_STAT_4	0x1E00020C	NONE	NO	NO
IPU_INT_STAT_5	0x1E000210	NONE	NO	NO
IPU_INT_STAT_6	0x1E000214	NONE	NO	NO
IPU_INT_STAT_7	0x1E000218	NONE	NO	NO
IPU_INT_STAT_8	0x1E00021C	NONE	NO	NO
IPU_INT_STAT_9	0x1E000220	NONE	NO	NO
IPU_INT_STAT_10	0x1E000224	NONE	NO	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
IPU_INT_STAT_11	0x1E000228	NONE	NO	NO
IPU_INT_STAT_12	0x1E00022C	NONE	NO	NO
IPU_INT_STAT_13	0x1E000230	NONE	NO	NO
IPU_INT_STAT_14	0x1E000234	NONE	NO	NO
IPU_INT_STAT_15	0x1E000238	NONE	NO	NO
IPU_CUR_BUF_0	0x1E00023C	NONE	NO	NO
IPU_CUR_BUF_1	0x1E000240	NONE	NO	NO
IPU_ALT_CUR_BUF_0	0x1E000244	NONE	NO	NO
IPU_ALT_CUR_BUF_1	0x1E000248	NONE	NO	NO
IPU_SRM_STAT	0x1E00024C	NONE	NO	NO
IPU_PROC_TASKS_STAT	0x1E000250	NONE	NO	NO
IPU_DISP_TASKS_STAT	0x1E000254	NONE	NO	NO
IPU_TRIPLE_CUR_BUF_0	0x1E000258	NONE	NO	NO
IPU_TRIPLE_CUR_BUF_1	0x1E00025C	NONE	NO	NO
IPU_TRIPLE_CUR_BUF_2	0x1E000260	NONE	NO	NO
IPU_TRIPLE_CUR_BUF_3	0x1E000264	NONE	NO	NO
IPU_CH_BUF0_RDY0	0x1E000268	NONE	NO	NO
IPU_CH_BUF0_RDY1	0x1E00026C	NONE	NO	NO
IPU_CH_BUF1_RDY0	0x1E000270	NONE	NO	NO
IPU_CH_BUF1_RDY1	0x1E000274	NONE	NO	NO
IPU_ALT_CH_BUF0_RDY0	0x1E000278	NONE	NO	NO
IPU_ALT_CH_BUF0_RDY1	0x1E00027C	NONE	NO	NO
IPU_ALT_CH_BUF1_RDY0	0x1E000280	NONE	NO	NO
IPU_ALT_CH_BUF1_RDY1	0x1E000284	NONE	NO	NO
IPU_CH_BUF2_RDY0	0x1E000288	NONE	NO	NO
IPU_CH_BUF2_RDY1	0x1E00028C	NONE	NO	NO
IDMAC_CONF	0x1E008000	NONE	YES	NO
IDMAC_CH_EN_1	0x1E008004	NONE	YES	NO
IDMAC_CH_EN_2	0x1E008008	NONE	YES	NO
IDMAC_SEP_ALPHA	0x1E00800C	NONE	YES	NO
IDMAC_ALT_SEP_ALPHA	0x1E008010	NONE	YES	NO
IDMAC_CH_PRI_1	0x1E008014	NONE	YES	NO
IDMAC_CH_PRI_2	0x1E008018	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
IDMAC_WM_EN_1	0x1E00801C	NONE	YES	NO
IDMAC_WM_EN_2	0x1E008020	NONE	YES	NO
IDMAC_LOCK_EN_1	0x1E008024	NONE	YES	NO
IDMAC_LOCK_EN_2	0x1E008028	NONE	YES	NO
IDMAC_SUB_ADDR_0	0x1E00802C	NONE	YES	NO
IDMAC_SUB_ADDR_1	0x1E008030	NONE	YES	NO
IDMAC_SUB_ADDR_2	0x1E008034	NONE	YES	NO
IDMAC_SUB_ADDR_3	0x1E008038	NONE	YES	NO
IDMAC_SUB_ADDR_4	0x1E00803C	NONE	YES	NO
IDMAC_BNDM_EN_1	0x1E008040	NONE	YES	NO
IDMAC_BNDM_EN_2	0x1E008044	NONE	YES	NO
IDMAC_SC_CORD	0x1E008048	NONE	YES	NO
IDMAC_SC_CORD1	0x1E00804C	NONE	YES	NO
IDMAC_CH_BUSY_1	0x1E008100	NONE	NO	NO
IDMAC_CH_BUSY_2	0x1E008104	NONE	NO	NO
DP_COM_CONF_SYNC	0x1F040000	0x1F040000	YES	YES
DP_GRAPH_WIND_CTRL_SYNC	0x1F040004	0x1F040004	YES	YES
DP_FG_POS_SYNC	0x1F040008	0x1F040008	YES	YES
DP_CUR_POS_SYNC	0x1F04000C	0x1F04000C	YES	YES
DP_CUR_MAP_SYNC	0x1F040010	0x1F040010	YES	YES
DP_GAMMA_C_SYNC_0	0x1F040014	0x1F040014	YES	YES
DP_GAMMA_C_SYNC_1	0x1F040018	0x1F040018	YES	YES
DP_GAMMA_C_SYNC_2	0x1F04001C	0x1F04001C	YES	YES
DP_GAMMA_C_SYNC_3	0x1F040020	0x1F040020	YES	YES
DP_GAMMA_C_SYNC_4	0x1F040024	0x1F040024	YES	YES
DP_GAMMA_C_SYNC_5	0x1F040028	0x1F040028	YES	YES
DP_GAMMA_C_SYNC_6	0x1F04002C	0x1F04002C	YES	YES
DP_GAMMA_C_SYNC_7	0x1F040030	0x1F040030	YES	YES
DP_GAMMA_S_SYNC_0	0x1F040034	0x1F040034	YES	YES
DP_GAMMA_S_SYNC_1	0x1F040038	0x1F040038	YES	YES
DP_GAMMA_S_SYNC_2	0x1F04003C	0x1F04003C	YES	YES
DP_GAMMA_S_SYNC_3	0x1F040040	0x1F040040	YES	YES
DP_CSCA_SYNC_0	0x1F040044	0x1F040044	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DP_CSCA_SYNC_1	0x1F040048	0x1F040048	YES	YES
DP_CSCA_SYNC_2	0x1F04004C	0x1F04004C	YES	YES
DP_CSCA_SYNC_3	0x1F040050	0x1F040050	YES	YES
DP_CSC_SYNC_0	0x1F040054	0x1F040054	YES	YES
DP_CSC_SYNC_1	0x1F040058	0x1F040058	YES	YES
DP_CUR_POS_ALT	0x1F04005C	0x1F04005C	YES	YES
DP_COM_CONF_ASYNC0	0x1F040060	0x1F040060	YES	YES
DP_GRAPH_WIND_CTRL_ASYNC0	0x1F040064	0x1F040064	YES	YES
DP_FG_POS_ASYNC0	0x1F040068	0x1F040068	YES	YES
DP_CUR_POS_ASYNC0	0x1F04006C	0x1F04006C	YES	YES
DP_CUR_MAP_ASYNC0	0x1F040070	0x1F040070	YES	YES
DP_GAMMA_C_ASYNC0_0	0x1F040074	0x1F040074	YES	YES
DP_GAMMA_C_ASYNC0_1	0x1F040078	0x1F040078	YES	YES
DP_GAMMA_C_ASYNC0_2	0x1F04007C	0x1F04007C	YES	YES
DP_GAMMA_C_ASYNC0_3	0x1F040080	0x1F040080	YES	YES
DP_GAMMA_C_ASYNC0_4	0x1F040084	0x1F040084	YES	YES
DP_GAMMA_C_ASYNC0_5	0x1F040088	0x1F040088	YES	YES
DP_GAMMA_C_ASYNC0_6	0x1F04008C	0x1F04008C	YES	YES
DP_GAMMA_C_ASYNC0_7	0x1F040090	0x1F040090	YES	YES
DP_GAMMA_S_ASYNC0_0	0x1F040094	0x1F040094	YES	YES
DP_GAMMA_S_ASYNC0_1	0x1F040098	0x1F040098	YES	YES
DP_GAMMA_S_ASYNC0_2	0x1F04009C	0x1F04009C	YES	YES
DP_GAMMA_S_ASYNC0_3	0x1F0400A0	0x1F0400A0	YES	YES
DP_CSCA_ASYNC0_0	0x1F0400A4	0x1F0400A4	YES	YES
DP_CSCA_ASYNC0_1	0x1F0400A8	0x1F0400A8	YES	YES
DP_CSCA_ASYNC0_2	0x1F0400AC	0x1F0400AC	YES	YES
DP_CSCA_ASYNC0_3	0x1F0400B0	0x1F0400B0	YES	YES
DP_CSC_ASYNC0_0	0x1F0400B4	0x1F0400B4	YES	YES
DP_CSC_ASYNC0_1	0x1F0400B8	0x1F0400B8	YES	YES
DP_COM_CONF_ASYNC1	0x1F0400BC	0x1F0400BC	YES	YES
DP_GRAPH_WIND_CTRL_ASYNC1	0x1F0400C0	0x1F0400C0	YES	YES
DP_FG_POS_ASYNC1	0x1F0400C4	0x1F0400C4	YES	YES
DP_CUR_POS_ASYNC1	0x1F0400C8	0x1F0400C8	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DP_CUR_MAP_ASYNC1	0x1F0400CC	0x1F0400CC	YES	YES
DP_GAMMA_C_ASYNC1_0	0x1F0400D0	0x1F0400D0	YES	YES
DP_GAMMA_C_ASYNC1_1	0x1F0400D4	0x1F0400D4	YES	YES
DP_GAMMA_C_ASYNC1_2	0x1F0400D8	0x1F0400D8	YES	YES
DP_GAMMA_C_ASYNC1_3	0x1F0400DC	0x1F0400DC	YES	YES
DP_GAMMA_C_ASYNC1_4	0x1F0400E0	0x1F0400E0	YES	YES
DP_GAMMA_C_ASYNC1_5	0x1F0400E4	0x1F0400E4	YES	YES
DP_GAMMA_C_ASYNC1_6	0x1F0400E8	0x1F0400E8	YES	YES
DP_GAMMA_C_ASYNC1_7	0x1F0400EC	0x1F0400EC	YES	YES
DP_GAMMA_S_ASYNC1_0	0x1F0400F0	0x1F0400F0	YES	YES
DP_GAMMA_S_ASYNC1_1	0x1F0400F4	0x1F0400F4	YES	YES
DP_GAMMA_S_ASYNC1_2	0x1F0400F8	0x1F0400F8	YES	YES
DP_GAMMA_S_ASYNC1_3	0x1F0400FC	0x1F0400FC	YES	YES
DP_CSCA_ASYNC1_0	0x1F040100	0x1F040100	YES	YES
DP_CSCA_ASYNC1_1	0x1F040104	0x1F040104	YES	YES
DP_CSCA_ASYNC1_2	0x1F040108	0x1F040108	YES	YES
DP_CSCA_ASYNC1_3	0x1F04010C	0x1F04010C	YES	YES
DP_CSC_ASYNC1_0	0x1F040110	0x1F040110	YES	YES
DP_CSC_ASYNC1_1	0x1F040114	0x1F040114	YES	YES
DP_DEBUG_CNT	0x1E0180BC	NONE	NO	NO
DP_DEBUG_STAT	0x1E0180C0	NONE	NO	NO
IC_CONF	0x1E020000	NONE	YES	NO
IC_PRP_ENC_RSC	0x1E020004	NONE	YES	NO
IC_PRP_VF_RSC	0x1E020008	NONE	YES	NO
IC_PP_RSC	0x1E02000C	NONE	YES	NO
IC_CMBP_1	0x1E020010	NONE	YES	NO
IC_CMBP_2	0x1E020014	NONE	YES	NO
IC_IDMAC_1	0x1E020018	NONE	YES	NO
IC_IDMAC_2	0x1E02001C	NONE	YES	NO
IC_IDMAC_3	0x1E020020	NONE	YES	NO
IC_IDMAC_4	0x1E020024	NONE	YES	NO
CSI0_SENS_CONF	0x1E030000	NONE	YES	NO
CSI0_SENS_FRM_SIZE	0x1E030004	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
CSI0_ACT_FRM_SIZE	0x1E030008	NONE	YES	NO
CSI0_OUT_FRM_CTRL	0x1E03000C	NONE	YES	NO
CSI0_TST_CTRL	0x1E030010	NONE	YES	NO
CSI0_CCIR_CODE_1	0x1E030014	NONE	YES	NO
CSI0_CCIR_CODE_2	0x1E030018	NONE	YES	NO
CSI0_CCIR_CODE_3	0x1E03001C	NONE	YES	NO
CSI0_DI	0x1E030020	NONE	YES	NO
CSI0_SKIP	0x1E030024	NONE	YES	NO
CSI0_CPD_CTRL	0x1E030028	0x1F040314	YES	NO
CSI0_CPD_RC_0	0x1E03002C	0x1F040318	YES	NO
CSI0_CPD_RC_1	0x1E030030	0x1F04031C	YES	NO
CSI0_CPD_RC_2	0x1E030034	0x1F040320	YES	NO
CSI0_CPD_RC_3	0x1E030038	0x1F040324	YES	NO
CSI0_CPD_RC_4	0x1E03003C	0x1F040328	YES	NO
CSI0_CPD_RC_5	0x1E030040	0x1F04032C	YES	NO
CSI0_CPD_RC_6	0x1E030044	0x1F040330	YES	NO
CSI0_CPD_RC_7	0x1E030048	0x1F040334	YES	NO
CSI0_CPD_RS_0	0x1E03004C	0x1F040338	YES	NO
CSI0_CPD_RS_1	0x1E030050	0x1F04033C	YES	NO
CSI0_CPD_RS_2	0x1E030054	0x1F040340	YES	NO
CSI0_CPD_RS_3	0x1E030058	0x1F040344	YES	NO
CSI0_CPD_GRC_0	0x1E03005C	0x1F040348	YES	NO
CSI0_CPD_GRC_1	0x1E030060	0x1F04034C	YES	NO
CSI0_CPD_GRC_2	0x1E030064	0x1F040350	YES	NO
CSI0_CPD_GRC_3	0x1E030068	0x1F040354	YES	NO
CSI0_CPD_GRC_4	0x1E03006C	0x1F040358	YES	NO
CSI0_CPD_GRC_5	0x1E030070	0x1F04035C	YES	NO
CSI0_CPD_GRC_6	0x1E030074	0x1F040360	YES	NO
CSI0_CPD_GRC_7	0x1E030078	0x1F040364	YES	NO
CSI0_CPD_GRS_0	0x1E03007C	0x1F040368	YES	NO
CSI0_CPD_GRS_1	0x1E030080	0x1F04036C	YES	NO
CSI0_CPD_GRS_2	0x1E030084	0x1F040370	YES	NO
CSI0_CPD_GRS_3	0x1E030088	0x1F040374	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
CSI0_CPD_GBC_0	0x1E03008C	0x1F040378	YES	NO
CSI0_CPD_GBC_1	0x1E030090	0x1F04037C	YES	NO
CSI0_CPD_GBC_2	0x1E030094	0x1F040380	YES	NO
CSI0_CPD_GBC_3	0x1E030098	0x1F040384	YES	NO
CSI0_CPD_GBC_4	0x1E03009C	0x1F040388	YES	NO
CSI0_CPD_GBC_5	0x1E0300A0	0x1F04038C	YES	NO
CSI0_CPD_GBC_6	0x1E0300A4	0x1F040390	YES	NO
CSI0_CPD_GBC_7	0x1E0300A8	0x1F040394	YES	NO
CSI0_CPD_GBS_0	0x1E0300AC	0x1F040398	YES	NO
CSI0_CPD_GBS_1	0x1E0300B0	0x1F04039C	YES	NO
CSI0_CPD_GBS_2	0x1E0300B4	0x1F0403A0	YES	NO
CSI0_CPD_GBS_3	0x1E0300B8	0x1F0403A4	YES	NO
CSI0_CPD_BC_0	0x1E0300BC	0x1F0403A8	YES	NO
CSI0_CPD_BC_1	0x1E0300C0	0x1F0403AC	YES	NO
CSI0_CPD_BC_2	0x1E0300C4	0x1F0403B0	YES	NO
CSI0_CPD_BC_3	0x1E0300C8	0x1F0403B4	YES	NO
CSI0_CPD_BC_4	0x1E0300CC	0x1F0403B8	YES	NO
CSI0_CPD_BC_5	0x1E0300D0	0x1F0403BC	YES	NO
CSI0_CPD_BC_6	0x1E0300D4	0x1F0403C0	YES	NO
CSI0_CPD_BC_7	0x1E0300D8	0x1F0403C4	YES	NO
CSI0_CPD_BS_0	0x1E0300DC	0x1F0403C8	YES	NO
CSI0_CPD_BS_1	0x1E0300E0	0x1F0403CC	YES	NO
CSI0_CPD_BS_2	0x1E0300E4	0x1F0403D0	YES	NO
CSI0_CPD_BS_3	0x1E0300E8	0x1F0403D4	YES	NO
CSI0_CPD_OFFSET1	0x1E0300EC	0x1F0403D8	YES	NO
CSI0_CPD_OFFSET2	0x1E0300F0	0x1F0403DC	YES	NO
CSI1_SENS_CONF	0x1E038000	NONE	YES	NO
CSI1_SENS_FRM_SIZE	0x1E038004	NONE	YES	NO
CSI1_ACT_FRM_SIZE	0x1E038008	NONE	YES	NO
CSI1_OUT_FRM_CTRL	0x1E03800C	NONE	YES	NO
CSI1_TST_CTRL	0x1E038010	NONE	YES	NO
CSI1_CCIR_CODE_1	0x1E038014	NONE	YES	NO
CSI1_CCIR_CODE_2	0x1E038018	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
CSI1_CCIR_CODE_3	0x1E03801C	NONE	YES	NO
CSI1_DI	0x1E038020	NONE	YES	NO
CSI1_SKIP	0x1E038024	NONE	YES	NO
CSI1_CPD_CTRL	0x1E038028	0x1F0403E0	YES	NO
CSI1_CPD_RC_0	0x1E03802C	0x1F0403E4	YES	NO
CSI1_CPD_RC_1	0x1E038030	0x1F0403E8	YES	NO
CSI1_CPD_RC_2	0x1E038034	0x1F0403EC	YES	NO
CSI1_CPD_RC_3	0x1E038038	0x1F0403F0	YES	NO
CSI1_CPD_RC_4	0x1E03803C	0x1F0403F4	YES	NO
CSI1_CPD_RC_5	0x1E038040	0x1F0403F8	YES	NO
CSI1_CPD_RC_6	0x1E038044	0x1F0403FC	YES	NO
CSI1_CPD_RC_7	0x1E038048	0x1F040400	YES	NO
CSI1_CPD_RS_0	0x1E03804C	0x1F040404	YES	NO
CSI1_CPD_RS_1	0x1E038050	0x1F040408	YES	NO
CSI1_CPD_RS_2	0x1E038054	0x1F04040C	YES	NO
CSI1_CPD_RS_3	0x1E038058	0x1F040410	YES	NO
CSI1_CPD_GRC_0	0x1E03805C	0x1F040414	YES	NO
CSI1_CPD_GRC_1	0x1E038060	0x1F040418	YES	NO
CSI1_CPD_GRC_2	0x1E038064	0x1F04041C	YES	NO
CSI1_CPD_GRC_3	0x1E038068	0x1F040420	YES	NO
CSI1_CPD_GRC_4	0x1E03806C	0x1F040424	YES	NO
CSI1_CPD_GRC_5	0x1E038070	0x1F040428	YES	NO
CSI1_CPD_GRC_6	0x1E038074	0x1F04042C	YES	NO
CSI1_CPD_GRC_7	0x1E038078	0x1F040430	YES	NO
CSI1_CPD_GRS_0	0x1E03807C	0x1F040434	YES	NO
CSI1_CPD_GRS_1	0x1E038080	0x1F040438	YES	NO
CSI1_CPD_GRS_2	0x1E038084	0x1F04043C	YES	NO
CSI1_CPD_GRS_3	0x1E038088	0x1F040440	YES	NO
CSI1_CPD_GBC_0	0x1E03808C	0x1F040444	YES	NO
CSI1_CPD_GBC_1	0x1E038090	0x1F040448	YES	NO
CSI1_CPD_GBC_2	0x1E038094	0x1F04044C	YES	NO
CSI1_CPD_GBC_3	0x1E038098	0x1F040450	YES	NO
CSI1_CPD_GBC_4	0x1E03809C	0x1F040454	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
CSI1_CPD_GBC_5	0x1E0380A0	0x1F040458	YES	NO
CSI1_CPD_GBC_6	0x1E0380A4	0x1F04045C	YES	NO
CSI1_CPD_GBC_7	0x1E0380A8	0x1F040460	YES	NO
CSI1_CPD_GBS_0	0x1E0380AC	0x1F040464	YES	NO
CSI1_CPD_GBS_1	0x1E0380B0	0x1F040468	YES	NO
CSI1_CPD_GBS_2	0x1E0380B4	0x1F04046C	YES	NO
CSI1_CPD_GBS_3	0x1E0380B8	0x1F040470	YES	NO
CSI1_CPD_BC_0	0x1E0380BC	0x1F040474	YES	NO
CSI1_CPD_BC_1	0x1E0380C0	0x1F040478	YES	NO
CSI1_CPD_BC_2	0x1E0380C4	0x1F04047C	YES	NO
CSI1_CPD_BC_3	0x1E0380C8	0x1F040480	YES	NO
CSI1_CPD_BC_4	0x1E0380CC	0x1F040484	YES	NO
CSI1_CPD_BC_5	0x1E0380D0	0x1F040488	YES	NO
CSI1_CPD_BC_6	0x1E0380D4	0x1F04048C	YES	NO
CSI1_CPD_BC_7	0x1E0380D8	0x1F040490	YES	NO
CSI1_CPD_BS_0	0x1E0380DC	0x1F040494	YES	NO
CSI1_CPD_BS_1	0x1E0380E0	0x1F040498	YES	NO
CSI1_CPD_BS_2	0x1E0380E4	0x1F04049C	YES	NO
CSI1_CPD_BS_3	0x1E0380E8	0x1F0404A0	YES	NO
CSI1_CPD_OFFSET1	0x1E0380EC	0x1F0404A4	YES	NO
CSI1_CPD_OFFSET2	0x1E0380F0	0x1F0404A8	YES	NO
DI0_GENERAL	0x1E040000	0x1F0404E4	YES	YES
DI0_BS_CLKGEN0	0x1E040004	0x1F0404E8	YES	YES
DI0_BS_CLKGEN1	0x1E040008	0x1F0404EC	YES	YES
DI0_SW_GEN0_1	0x1E04000C	0x1F0404F0	YES	YES
DI0_SW_GEN0_2	0x1E040010	0x1F0404F4	YES	YES
DI0_SW_GEN0_3	0x1E040014	0x1F0404F8	YES	YES
DI0_SW_GEN0_4	0x1E040018	0x1F0404FC	YES	YES
DI0_SW_GEN0_5	0x1E04001C	0x1F040500	YES	YES
DI0_SW_GEN0_6	0x1E040020	0x1F040504	YES	YES
DI0_SW_GEN0_7	0x1E040024	0x1F040508	YES	YES
DI0_SW_GEN0_8	0x1E040028	0x1F04050C	YES	YES
DI0_SW_GEN0_9	0x1E04002C	0x1F040510	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI0_SW_GEN1_1	0x1E040030	0x1F040514	YES	YES
DI0_SW_GEN1_2	0x1E040034	0x1F040518	YES	YES
DI0_SW_GEN1_3	0x1E040038	0x1F04051C	YES	YES
DI0_SW_GEN1_4	0x1E04003C	0x1F040520	YES	YES
DI0_SW_GEN1_5	0x1E040040	0x1F040524	YES	YES
DI0_SW_GEN1_6	0x1E040044	0x1F040528	YES	YES
DI0_SW_GEN1_7	0x1E040048	0x1F04052C	YES	YES
DI0_SW_GEN1_8	0x1E04004C	0x1F040530	YES	YES
DI0_SW_GEN1_9	0x1E040050	0x1F040534	YES	YES
DI0_SYNC_AS_GEN	0x1E040054	0x1F040538	YES	YES
DI0_DW_GEN_0	0x1E040058	0x1F04053C	YES	YES
DI0_DW_GEN_1	0x1E04005C	0x1F040540	YES	YES
DI0_DW_GEN_2	0x1E040060	0x1F040544	YES	YES
DI0_DW_GEN_3	0x1E040064	0x1F040548	YES	YES
DI0_DW_GEN_4	0x1E040068	0x1F04054C	YES	YES
DI0_DW_GEN_5	0x1E04006C	0x1F040550	YES	YES
DI0_DW_GEN_6	0x1E040070	0x1F040554	YES	YES
DI0_DW_GEN_7	0x1E040074	0x1F040558	YES	YES
DI0_DW_GEN_8	0x1E040078	0x1F04055C	YES	YES
DI0_DW_GEN_9	0x1E04007C	0x1F040560	YES	YES
DI0_DW_GEN_10	0x1E040080	0x1F040564	YES	YES
DI0_DW_GEN_11	0x1E040084	0x1F040568	YES	YES
DI0_DW_SET0_0	0x1E040088	0x1F04056C	YES	YES
DI0_DW_SET0_1	0x1E04008C	0x1F040570	YES	YES
DI0_DW_SET0_2	0x1E040090	0x1F040574	YES	YES
DI0_DW_SET0_3	0x1E040094	0x1F040578	YES	YES
DI0_DW_SET0_4	0x1E040098	0x1F04057C	YES	YES
DI0_DW_SET0_5	0x1E04009C	0x1F040580	YES	YES
DI0_DW_SET0_6	0x1E0400A0	0x1F040584	YES	YES
DI0_DW_SET0_7	0x1E0400A4	0x1F040588	YES	YES
DI0_DW_SET0_8	0x1E0400A8	0x1F04058C	YES	YES
DI0_DW_SET0_9	0x1E0400AC	0x1F040590	YES	YES
DI0_DW_SET0_10	0x1E0400B0	0x1F040594	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI0_DW_SET0_11	0x1E0400B4	0x1F040598	YES	YES
DI0_DW_SET1_0	0x1E0400B8	0x1F04059C	YES	YES
DI0_DW_SET1_1	0x1E0400BC	0x1F0405A0	YES	YES
DI0_DW_SET1_2	0x1E0400C0	0x1F0405A4	YES	YES
DI0_DW_SET1_3	0x1E0400C4	0x1F0405A8	YES	YES
DI0_DW_SET1_4	0x1E0400C8	0x1F0405AC	YES	YES
DI0_DW_SET1_5	0x1E0400CC	0x1F0405B0	YES	YES
DI0_DW_SET1_6	0x1E0400D0	0x1F0405B4	YES	YES
DI0_DW_SET1_7	0x1E0400D4	0x1F0405B8	YES	YES
DI0_DW_SET1_8	0x1E0400D8	0x1F0405BC	YES	YES
DI0_DW_SET1_9	0x1E0400DC	0x1F0405C0	YES	YES
DI0_DW_SET1_10	0x1E0400E0	0x1F0405C4	YES	YES
DI0_DW_SET1_11	0x1E0400E4	0x1F0405C8	YES	YES
DI0_DW_SET2_0	0x1E0400E8	0x1F0405CC	YES	YES
DI0_DW_SET2_1	0x1E0400EC	0x1F0405D0	YES	YES
DI0_DW_SET2_2	0x1E0400F0	0x1F0405D4	YES	YES
DI0_DW_SET2_3	0x1E0400F4	0x1F0405D8	YES	YES
DI0_DW_SET2_4	0x1E0400F8	0x1F0405DC	YES	YES
DI0_DW_SET2_5	0x1E0400FC	0x1F0405E0	YES	YES
DI0_DW_SET2_6	0x1E040100	0x1F0405E4	YES	YES
DI0_DW_SET2_7	0x1E040104	0x1F0405E8	YES	YES
DI0_DW_SET2_8	0x1E040108	0x1F0405EC	YES	YES
DI0_DW_SET2_9	0x1E04010C	0x1F0405F0	YES	YES
DI0_DW_SET2_10	0x1E040110	0x1F0405F4	YES	YES
DI0_DW_SET2_11	0x1E040114	0x1F0405F8	YES	YES
DI0_DW_SET3_0	0x1E040118	0x1F0405FC	YES	YES
DI0_DW_SET3_1	0x1E04011C	0x1F040600	YES	YES
DI0_DW_SET3_2	0x1E040120	0x1F040604	YES	YES
DI0_DW_SET3_3	0x1E040124	0x1F040608	YES	YES
DI0_DW_SET3_4	0x1E040128	0x1F04060C	YES	YES
DI0_DW_SET3_5	0x1E04012C	0x1F040610	YES	YES
DI0_DW_SET3_6	0x1E040130	0x1F040614	YES	YES
DI0_DW_SET3_7	0x1E040134	0x1F040618	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI0_DW_SET3_8	0x1E040138	0x1F04061C	YES	YES
DI0_DW_SET3_9	0x1E04013C	0x1F040620	YES	YES
DI0_DW_SET3_10	0x1E040140	0x1F040624	YES	YES
DI0_DW_SET3_11	0x1E040144	0x1F040628	YES	YES
DI0_STP_REP_1	0x1E040148	0x1F04062C	YES	YES
DI0_STP_REP_2	0x1E04014C	0x1F040630	YES	YES
DI0_STP_REP_3	0x1E040150	0x1F040634	YES	YES
DI0_STP_REP_4	0x1E040154	0x1F040638	YES	YES
DI0_STP_REP_9	0x1E040158	0x1F04063C	YES	YES
DI0_SER_CONF	0x1E04015C	0x1F040640	YES	YES
DI0_SSC	0x1E040160	0x1F040644	YES	YES
DI0_POL	0x1E040164	0x1F040648	YES	YES
DI0_AW0	0x1E040168	0x1F04064C	YES	YES
DI0_AW1	0x1E04016C	0x1F040650	YES	YES
DI0_SCR_CONF	0x1E040170	0x1F040654	YES	YES
DI0_STAT	0x1E040174	NONE	NO	NO
DI1_GENERAL	0x1E048000	0x1F040658	YES	YES
DI1_BS_CLKGEN0	0x1E048004	0x1F04065C	YES	YES
DI1_BS_CLKGEN1	0x1E048008	0x1F040660	YES	YES
DI1_SW_GEN0_1	0x1E04800C	0x1F040664	YES	YES
DI1_SW_GEN0_2	0x1E048010	0x1F040668	YES	YES
DI1_SW_GEN0_3	0x1E048014	0x1F04066C	YES	YES
DI1_SW_GEN0_4	0x1E048018	0x1F040670	YES	YES
DI1_SW_GEN0_5	0x1E04801C	0x1F040674	YES	YES
DI1_SW_GEN0_6	0x1E048020	0x1F040678	YES	YES
DI1_SW_GEN0_7	0x1E048024	0x1F04067C	YES	YES
DI1_SW_GEN0_8	0x1E048028	0x1F040680	YES	YES
DI1_SW_GEN0_9	0x1E04802C	0x1F040684	YES	YES
DI1_SW_GEN1_1	0x1E048030	0x1F040688	YES	YES
DI1_SW_GEN1_2	0x1E048034	0x1F04068C	YES	YES
DI1_SW_GEN1_3	0x1E048038	0x1F040690	YES	YES
DI1_SW_GEN1_4	0x1E04803C	0x1F040694	YES	YES
DI1_SW_GEN1_5	0x1E048040	0x1F040698	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI1_SW_GEN1_6	0x1E048044	0x1F04069C	YES	YES
DI1_SW_GEN1_7	0x1E048048	0x1F0406A0	YES	YES
DI1_SW_GEN1_8	0x1E04804C	0x1F0406A4	YES	YES
DI1_SW_GEN1_9	0x1E048050	0x1F0406A8	YES	YES
DI1_SYNC_AS_GEN	0x1E048054	0x1F0406AC	YES	YES
DI1_DW_GEN_0	0x1E048058	0x1F0406B0	YES	YES
DI1_DW_GEN_1	0x1E04805C	0x1F0406B4	YES	YES
DI1_DW_GEN_2	0x1E048060	0x1F0406B8	YES	YES
DI1_DW_GEN_3	0x1E048064	0x1F0406BC	YES	YES
DI1_DW_GEN_4	0x1E048068	0x1F0406C0	YES	YES
DI1_DW_GEN_5	0x1E04806C	0x1F0406C4	YES	YES
DI1_DW_GEN_6	0x1E048070	0x1F0406C8	YES	YES
DI1_DW_GEN_7	0x1E048074	0x1F0406CC	YES	YES
DI1_DW_GEN_8	0x1E048078	0x1F0406D0	YES	YES
DI1_DW_GEN_9	0x1E04807C	0x1F0406D4	YES	YES
DI1_DW_GEN_10	0x1E048080	0x1F0406D8	YES	YES
DI1_DW_GEN_11	0x1E048084	0x1F0406DC	YES	YES
DI1_DW_SET0_0	0x1E048088	0x1F0406E0	YES	YES
DI1_DW_SET0_1	0x1E04808C	0x1F0406E4	YES	YES
DI1_DW_SET0_2	0x1E048090	0x1F0406E8	YES	YES
DI1_DW_SET0_3	0x1E048094	0x1F0406EC	YES	YES
DI1_DW_SET0_4	0x1E048098	0x1F0406F0	YES	YES
DI1_DW_SET0_5	0x1E04809C	0x1F0406F4	YES	YES
DI1_DW_SET0_6	0x1E0480A0	0x1F0406F8	YES	YES
DI1_DW_SET0_7	0x1E0480A4	0x1F0406FC	YES	YES
DI1_DW_SET0_8	0x1E0480A8	0x1F040700	YES	YES
DI1_DW_SET0_9	0x1E0480AC	0x1F040704	YES	YES
DI1_DW_SET0_10	0x1E0480B0	0x1F040708	YES	YES
DI1_DW_SET0_11	0x1E0480B4	0x1F04070C	YES	YES
DI1_DW_SET1_0	0x1E0480B8	0x1F040710	YES	YES
DI1_DW_SET1_1	0x1E0480BC	0x1F040714	YES	YES
DI1_DW_SET1_2	0x1E0480C0	0x1F040718	YES	YES
DI1_DW_SET1_3	0x1E0480C4	0x1F04071C	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI1_DW_SET1_4	0x1E0480C8	0x1F040720	YES	YES
DI1_DW_SET1_5	0x1E0480CC	0x1F040724	YES	YES
DI1_DW_SET1_6	0x1E0480D0	0x1F040728	YES	YES
DI1_DW_SET1_7	0x1E0480D4	0x1F04072C	YES	YES
DI1_DW_SET1_8	0x1E0480D8	0x1F040730	YES	YES
DI1_DW_SET1_9	0x1E0480DC	0x1F040734	YES	YES
DI1_DW_SET1_10	0x1E0480E0	0x1F040738	YES	YES
DI1_DW_SET1_11	0x1E0480E4	0x1F04073C	YES	YES
DI1_DW_SET2_0	0x1E0480E8	0x1F040740	YES	YES
DI1_DW_SET2_1	0x1E0480EC	0x1F040744	YES	YES
DI1_DW_SET2_2	0x1E0480F0	0x1F040748	YES	YES
DI1_DW_SET2_3	0x1E0480F4	0x1F04074C	YES	YES
DI1_DW_SET2_4	0x1E0480F8	0x1F040750	YES	YES
DI1_DW_SET2_5	0x1E0480FC	0x1F040754	YES	YES
DI1_DW_SET2_6	0x1E048100	0x1F040758	YES	YES
DI1_DW_SET2_7	0x1E048104	0x1F04075C	YES	YES
DI1_DW_SET2_8	0x1E048108	0x1F040760	YES	YES
DI1_DW_SET2_9	0x1E04810C	0x1F040764	YES	YES
DI1_DW_SET2_10	0x1E048110	0x1F040768	YES	YES
DI1_DW_SET2_11	0x1E048114	0x1F04076C	YES	YES
DI1_DW_SET3_0	0x1E048118	0x1F040770	YES	YES
DI1_DW_SET3_1	0x1E04811C	0x1F040774	YES	YES
DI1_DW_SET3_2	0x1E048120	0x1F040778	YES	YES
DI1_DW_SET3_3	0x1E048124	0x1F04077C	YES	YES
DI1_DW_SET3_4	0x1E048128	0x1F040780	YES	YES
DI1_DW_SET3_5	0x1E04812C	0x1F040784	YES	YES
DI1_DW_SET3_6	0x1E048130	0x1F040788	YES	YES
DI1_DW_SET3_7	0x1E048134	0x1F04078C	YES	YES
DI1_DW_SET3_8	0x1E048138	0x1F040790	YES	YES
DI1_DW_SET3_9	0x1E04813C	0x1F040794	YES	YES
DI1_DW_SET3_10	0x1E048140	0x1F040798	YES	YES
DI1_DW_SET3_11	0x1E048144	0x1F04079C	YES	YES
DI1_STP_REP_1	0x1E048148	0x1F0407A0	YES	YES

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DI1_STP_REP_2	0x1E04814C	0x1F0407A4	YES	YES
DI1_STP_REP_3	0x1E048150	0x1F0407A8	YES	YES
DI1_STP_REP_4	0x1E048154	0x1F0407AC	YES	YES
DI1_STP_REP_9	0x1E048158	0x1F0407B0	YES	YES
DI1_SER_CONF	0x1E04815C	0x1F0407B4	YES	YES
DI1_SSC	0x1E048160	0x1F0407B8	YES	YES
DI1_POL	0x1E048164	0x1F0407BC	YES	YES
DI1_AW0	0x1E048168	0x1F0407C0	YES	YES
DI1_AW1	0x1E04816C	0x1F0407C4	YES	YES
DI1_SCR_CONF	0x1E048170	0x1F0407C8	YES	YES
DI1_STAT	0x1E048174	NONE	NO	NO
SMFC_MAP	0x1E050000	NONE	YES	NO
SMFC_WMC	0x1E050004	NONE	YES	NO
SMFC_BS	0x1E050008	NONE	YES	NO
DC_READ_CH_CONF	0x1E058000	NONE	YES	NO
DC_READ_CH_ADDR	0x1E058004	NONE	YES	NO
DC_RL0_CH_0	0x1E058008	NONE	YES	NO
DC_RL1_CH_0	0x1E05800C	NONE	YES	NO
DC_RL2_CH_0	0x1E058010	NONE	YES	NO
DC_RL3_CH_0	0x1E058014	NONE	YES	NO
DC_RL4_CH_0	0x1E058018	NONE	YES	NO
DC_WR_CH_CONF_1	0x1E05801C	NONE	YES	NO
DC_WR_CH_ADDR_1	0x1E058020	NONE	YES	NO
DC_RL0_CH_1	0x1E058024	NONE	YES	NO
DC_RL1_CH_1	0x1E058028	NONE	YES	NO
DC_RL2_CH_1	0x1E05802C	NONE	YES	NO
DC_RL3_CH_1	0x1E058030	NONE	YES	NO
DC_RL4_CH_1	0x1E058034	NONE	YES	NO
DC_WR_CH_CONF_2	0x1E058038	0x1F0404AC	YES	NO
DC_WR_CH_ADDR_2	0x1E05803C	0x1F0404B0	YES	NO
DC_RL0_CH_2	0x1E058040	0x1F0404B4	YES	NO
DC_RL1_CH_2	0x1E058044	0x1F0404B8	YES	NO
DC_RL2_CH_2	0x1E058048	0x1F0404BC	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DC_RL3_CH_2	0x1E05804C	0x1F0404C0	YES	NO
DC_RL4_CH_2	0x1E058050	0x1F0404C4	YES	NO
DC_CMD_CH_CONF_3	0x1E058054	NONE	YES	NO
DC_CMD_CH_CONF_4	0x1E058058	NONE	YES	NO
DC_WR_CH_CONF_5	0x1E05805C	NONE	YES	NO
DC_WR_CH_ADDR_5	0x1E058060	NONE	YES	NO
DC_RL0_CH_5	0x1E058064	NONE	YES	NO
DC_RL1_CH_5	0x1E058068	NONE	YES	NO
DC_RL2_CH_5	0x1E05806C	NONE	YES	NO
DC_RL3_CH_5	0x1E058070	NONE	YES	NO
DC_RL4_CH_5	0x1E058074	NONE	YES	NO
DC_WR_CH_CONF_6	0x1E058078	0x1F0404C8	YES	NO
DC_WR_CH_ADDR_6	0x1E05807C	0x1F0404CC	YES	NO
DC_RL0_CH_6	0x1E058080	0x1F0404D0	YES	NO
DC_RL1_CH_6	0x1E058084	0x1F0404D4	YES	NO
DC_RL2_CH_6	0x1E058088	0x1F0404D8	YES	NO
DC_RL3_CH_6	0x1E05808C	0x1F0404DC	YES	NO
DC_RL4_CH_6	0x1E058090	0x1F0404E0	YES	NO
DC_WR_CH_CONF1_8	0x1E058094	NONE	YES	NO
DC_WR_CH_CONF2_8	0x1E058098	NONE	YES	NO
DC_RL1_CH_8	0x1E05809C	NONE	YES	NO
DC_RL2_CH_8	0x1E0580A0	NONE	YES	NO
DC_RL3_CH_8	0x1E0580A4	NONE	YES	NO
DC_RL4_CH_8	0x1E0580A8	NONE	YES	NO
DC_RL5_CH_8	0x1E0580AC	NONE	YES	NO
DC_RL6_CH_8	0x1E0580B0	NONE	YES	NO
DC_WR_CH_CONF1_9	0x1E0580B4	NONE	YES	NO
DC_WR_CH_CONF2_9	0x1E0580B8	NONE	YES	NO
DC_RL1_CH_9	0x1E0580BC	NONE	YES	NO
DC_RL2_CH_9	0x1E0580C0	NONE	YES	NO
DC_RL3_CH_9	0x1E0580C4	NONE	YES	NO
DC_RL4_CH_9	0x1E0580C8	NONE	YES	NO
DC_RL5_CH_9	0x1E0580CC	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DC_RL6_CH_9	0x1E0580D0	NONE	YES	NO
DC_GEN	0x1E0580D4	NONE	YES	NO
DC_DISP_CONF1_0	0x1E0580D8	NONE	YES	NO
DC_DISP_CONF1_0_BE_L_INC_0	0x1E0580D8	NONE	YES	NO
DC_DISP_CONF1_0_INCREMENT_0	0x1E0580D8	NONE	YES	NO
DC_DISP_CONF1_1	0x1E0580DC	NONE	YES	NO
DC_DISP_CONF1_1_BE_L_INC_1	0x1E0580DC	NONE	YES	NO
DC_DISP_CONF1_1_INCREMENT_1	0x1E0580DC	NONE	YES	NO
DC_DISP_CONF1_2	0x1E0580E0	NONE	YES	NO
DC_DISP_CONF1_2_BE_L_INC_2	0x1E0580E0	NONE	YES	NO
DC_DISP_CONF1_2_INCREMENT_2	0x1E0580E0	NONE	YES	NO
DC_DISP_CONF1_3	0x1E0580E4	NONE	YES	NO
DC_DISP_CONF1_3_BE_L_INC_3	0x1E0580E4	NONE	YES	NO
DC_DISP_CONF1_3_INCREMENT_3	0x1E0580E4	NONE	YES	NO
DC_DISP_CONF2_0	0x1E0580E8	NONE	YES	NO
DC_DISP_CONF2_1	0x1E0580EC	NONE	YES	NO
DC_DISP_CONF2_2	0x1E0580F0	NONE	YES	NO
DC_DISP_CONF2_3	0x1E0580F4	NONE	YES	NO
DC_DI0_CONF_1	0x1E0580F8	NONE	YES	NO
DC_DI0_CONF_2	0x1E0580FC	NONE	YES	NO
DC_DI1_CONF_1	0x1E058100	NONE	YES	NO
DC_DI1_CONF_2	0x1E058104	NONE	YES	NO
DC_MAP_CONF_0	0x1E058108	NONE	YES	NO
DC_MAP_CONF_1	0x1E05810C	NONE	YES	NO
DC_MAP_CONF_2	0x1E058110	NONE	YES	NO
DC_MAP_CONF_3	0x1E058114	NONE	YES	NO
DC_MAP_CONF_4	0x1E058118	NONE	YES	NO
DC_MAP_CONF_5	0x1E05811C	NONE	YES	NO
DC_MAP_CONF_6	0x1E058120	NONE	YES	NO
DC_MAP_CONF_7	0x1E058124	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DC_MAP_CONF_8	0x1E058128	NONE	YES	NO
DC_MAP_CONF_9	0x1E05812C	NONE	YES	NO
DC_MAP_CONF_10	0x1E058130	NONE	YES	NO
DC_MAP_CONF_11	0x1E058134	NONE	YES	NO
DC_MAP_CONF_12	0x1E058138	NONE	YES	NO
DC_MAP_CONF_13	0x1E05813C	NONE	YES	NO
DC_MAP_CONF_14	0x1E058140	NONE	YES	NO
DC_MAP_CONF_15	0x1E058144	NONE	YES	NO
DC_MAP_CONF_16	0x1E058148	NONE	YES	NO
DC_MAP_CONF_17	0x1E05814C	NONE	YES	NO
DC_MAP_CONF_18	0x1E058150	NONE	YES	NO
DC_MAP_CONF_19	0x1E058154	NONE	YES	NO
DC_MAP_CONF_20	0x1E058158	NONE	YES	NO
DC_MAP_CONF_21	0x1E05815C	NONE	YES	NO
DC_MAP_CONF_22	0x1E058160	NONE	YES	NO
DC_MAP_CONF_23	0x1E058164	NONE	YES	NO
DC_MAP_CONF_24	0x1E058168	NONE	YES	NO
DC_MAP_CONF_25	0x1E05816C	NONE	YES	NO
DC_MAP_CONF_26	0x1E058170	NONE	YES	NO
DC_UGDE0_0	0x1E058174	NONE	YES	NO
DC_UGDE0_1	0x1E058178	NONE	YES	NO
DC_UGDE0_2	0x1E05817C	NONE	YES	NO
DC_UGDE0_3	0x1E058180	NONE	YES	NO
DC_UGDE1_0	0x1E058184	NONE	YES	NO
DC_UGDE1_1	0x1E058188	NONE	YES	NO
DC_UGDE1_2	0x1E05818C	NONE	YES	NO
DC_UGDE1_3	0x1E058190	NONE	YES	NO
DC_UGDE2_0	0x1E058194	NONE	YES	NO
DC_UGDE2_1	0x1E058198	NONE	YES	NO
DC_UGDE2_2	0x1E05819C	NONE	YES	NO
DC_UGDE2_3	0x1E0581A0	NONE	YES	NO
DC_UGDE3_0	0x1E0581A4	NONE	YES	NO
DC_UGDE3_1	0x1E0581A8	NONE	YES	NO

Table 42-436. IPUv3EX SRM entries mapping (continued)

Register's name	Register's address	SRM entry	PG	LPSR
DC_UGDE3_2	0x1E0581AC	NONE	YES	NO
DC_UGDE3_3	0x1E0581B0	NONE	YES	NO
DC_LLA0	0x1E0581B4	NONE	YES	NO
DC_LLA1	0x1E0581B8	NONE	YES	NO
DC_R_LLA0	0x1E0581BC	NONE	YES	NO
DC_R_LLA1	0x1E0581C0	NONE	YES	NO
DC_WR_CH_ADDR_5_ALT	0x1E0581C4	NONE	YES	NO
DC_STAT	0x1E0581C8	NONE	NO	NO
DMFC_RD_CHAN	0x1E060000	NONE	YES	NO
DMFC_WR_CHAN	0x1E060004	NONE	YES	NO
DMFC_WR_CHAN_DEF	0x1E060008	NONE	YES	NO
DMFC_DP_CHAN	0x1E06000C	NONE	YES	NO
DMFC_DP_CHAN_DEF	0x1E060010	NONE	YES	NO
DMFC_GENERAL1	0x1E060014	NONE	YES	NO
DMFC_GENERAL2	0x1E060018	NONE	YES	NO
DMFC_IC_CTRL	0x1E06001C	NONE	YES	NO
DMFC_WR_CHAN_ALT	0x1E060020	NONE	YES	NO
DMFC_WR_CHAN_DEF_ALT	0x1E060024	NONE	YES	NO
DMFC_DP_CHAN_ALT	0x1E060028	NONE	YES	NO
DMFC_DP_CHAN_DEF_ALT	0x1E06002C	NONE	YES	NO
DMFC_GENERAL1_ALT	0x1E060030	NONE	YES	NO
DMFC_STAT	0x1E060034	NONE	NO	NO
VDI_FSIZE	0x1E068000	NONE	YES	NO
VDI_C	0x1E068004	NONE	YES	NO

42.3.12.7 Memory Access Unit

The Memory Access Unit (MA) supports MCU access to the IPUv3EX internal memories. Some of the IPUv3EX

internal memories are memory mapped. This unit handles accessing these memories. The table below describe the accessible memories and their limitations.

Table 42-437. Internal Memories Access Support and Limitations

Memory	Function	Support and Limitations
lut	IDMAC's look up table	Accessible only when All the IDMAC channels that use the LUT are disabled
cpmem	IDMAC's Channel parameter Memory	This memory can be accessed while tasks are enabled. Must be configured before enabling processing tasks. IDMAC channel parameters must not be changed in the CPM when the corresponding DMA channel is enabled excluding the base addresses (EBA0 and EBA1). One of these parameters can be changed during channel operation if it relates to the non-active double buffer.
tpm	IC's task parameter memory	Must be configured before enabling processing tasks. This memory can be accessed while tasks are enabled. IC task parameters must not be changed in the TPM when the corresponding task is active.
dc_template	DC's template memory	Can be configured before enabling tasks. Must not be accessed while operation. Both read and write access to the DC's template memory are forbidden when there is any enabled channel which can use the template

42.3.12.8 SISG - Still Image Synchronization Generator

The IPUv3EX incudes a "Still Image Synchronization Generator" (SISG), providing time-sensitive control signals synchronizing the image sensor with camera peripherals, such as a flash lamp and a mechanical shutter.

The SISG is implemented using a single time base counter, and six Time Compare Units - as described in Figure 42-463.

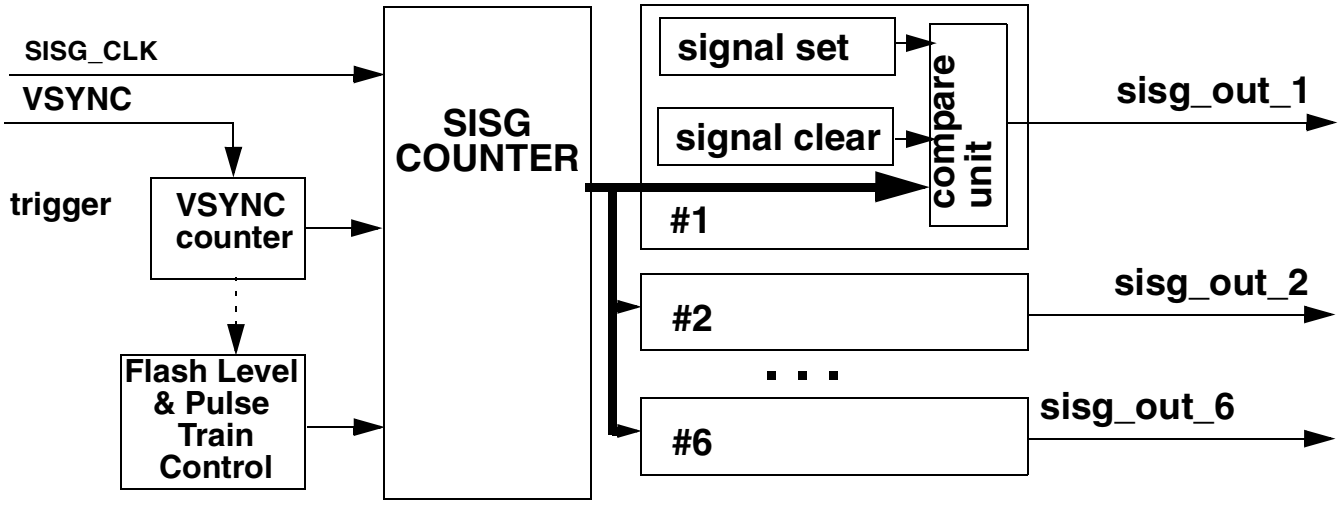


Figure 42-463. Still Image Signal Generator

The SISG inputs are

- Activation trigger
- VSYNC: the frame-boundary signal from the sensor.

The SISG is activated by one of the following triggers

- The MCU by setting the MCU_ACTV_TRIG bit
- An external signal (GPIO pin).

Upon activation, the counter is reset and then there are the following two possibilities

- Counting starts immediately
- Counting start is delayed until one of the next 7 VSYNC signals (programmable via the NO_OF_VSYNC bits).

During the counting period, the SISG can generate up to 6 output strobes.

- Each strobe can be individually enabled or disabled and has a programmable polarity
- The edges of the strobes are generated at specified counter values - to achieve pixel-level resolution - as specified by programmable SISG_SET& SISG_CLR time tag registers.
- The clock has 25 bits, to allow strobe generation during a time period of up to two 12M pixel frames

The SISG can repeat the above sequence for up to 32 cycles (this is provided to generate a train of flash pulses, for anti-red-eye or for measurements in low-light conditions). The repetition is implemented by resetting the counter and this can be triggered by one of the following events:

- A VSYNC signal
- A pre-defined value reached by the counter.

After the last sequence, when the counter reaches its maximal value, it stops counting and the SISG remains in idle mode until the next activation.

42.3.12.9 Clock Change procedure

The IPUv3EX supports dynamic clock rate changes.

Types of change:

- DVFS transitions: frequent, initiated by the SoC's power modes controller (GPC) and the SoC's clock controller module (CCM)
- Other: infrequent, initiated by SW.

IPUv3EX may have on-going activities at this stage.

The display interface clocks may either change or remain unchanged. During screen refresh, the display clock would typically not change. During asynchronous access, it may be appropriate to change also the display interface clock. The choice between these options would be made in advance by the user

If the IPUv3EX display interface uses the external clock (ipp_di_0_ext_clk or ipp_di_1_ext_clk) source, it remains unchanged. A change in the rate of this clock is performed fully by SW, without the special HW support described below. In particular, the SW may have to stop explicitly any interaction with the display (e.g. screen refresh) before performing the change.

The user is responsible to make sure that the lowest planned clock (in DVFS transitions) is still high enough to support the expected activities (e.g. data rate through the display bus)

The procedure below describes the IPUv3EX handshaking with the CCM.

1. The user prepares 2 sets of clock modes CLOCK_MODE_0 is the default clock mode, CLOCK_MODE_1 is the alternate clock mode. The IPUv3EX toggles between these two settings following the next assertion of ipg_clk_change_rq. If the user sets the SRM_CLOCK_CHANGE_MODE bit then he should also prepare the registers in the SRM for each of the DIs. These registers include all the DI settings adjusted to the new clock.
2. CCM asserts the ipg_clk_change_rq signal when a clock change is needed
3. The CM calculates the new clock frequency and send it to the DI (signals are di0_clk_freq, and di1_clk_freq). These signals should be sent to the DI only after getting the di_clk_change_ack signal from the DIs. The values of this field could be.
    - 00 - 1/4 of full frequency
    - 01 - 1/2 of full frequency
    - 10 - full frequency
    - 11 - illegal
4. The CM sends a cm_clk_change_rq signal to the DIs.
5. The DI stops the clock to the display (freeze mode) according to DI0_CLOCK_STOP_MODE & DI1_CLOCK_STOP_MODE bits. If the DI is disable, the ACK from the DI is not needed and the CM will assume that the DI sent an ACK.

6. Once the clock to the display is stopped, the DI sends a signal to the CM called di_clk_change_ack
7. The CM wait for the clock change signals from both of the DIs
8. If the SRM_CLOCK_CHANGE_MODE bit is set the CM should read the new DI settings from the SRM and override the previous DI settings. Then the CM clears the SRM_CLOCK_CHANGE_MODE bit
9. Once the above is complete the CM asserts the ipg_clk_change_ack signal to the CCM
10. The CM sends the signals di0_clk_freq, and di1_clk_freq to the DIs.
11. The CCM will negate the cm_clk_change_rq
12. The CCM will now change the clocks
13. When the new clock arrives the ipu_clk_changed signal will be asserted.
14. The state machine on each DI will now move out of freeze mode and continue working with the new clock.

42.3.12.10 Low Power Modes - Stop, PG and LPSR modes

IPUv3EX supports 3 low power modes

- STOP: on this mode the clock to the IPUv3EX is stopped
- PG: power gating, where the clock to the IPUv3EX is stopped and the power is for the IPUv3EX is turned off. Some of the memories are kept powered, so the wake up from this mode will be faster.
- LPSR: low power screen refresh.The clock to the IPUv3EX is changed to a slower frequency, the IPUv3EX performs only screen refresh.

The user should not request both LPSR and PG. at the same time. This case is not supported and the results are not predictable.

The CCM may assert one of the 2 signals: stop_clk_at_stop_req OR stop_clk_at_wait_req

The IPUv3EX OR them internally as there's no difference between them with regards to IPUv3EX's behavior.

In all these modes the clock to the IPUv3EX is going to be stopped (assertion of stop_clk_at_stop_req).

IPUv3EX should complete all his tasks:

- CSIs complete transferring the last frame. Wait for csi_busy = 0
- IDMAC completes all the flows (all CH_BUSY = 0)
- All the flows in the FSU are complete. New ones do not start.
- CM sends stop request to the DIs
- Once the DI sent all the data to the display, it asserts an ACK signal

Only when all the above occurs, the CM can assert an internal signal called "IPU_IDLE"

IPU_IDLE is the starting point for any of the low power modes.

42.3.12.10.1 STOP Mode

In this mode the IPUv3EX sends the ipu_stby_ack after getting to IPU_IDLE state. The CCM will gate off the clock to the IPUv3EX.

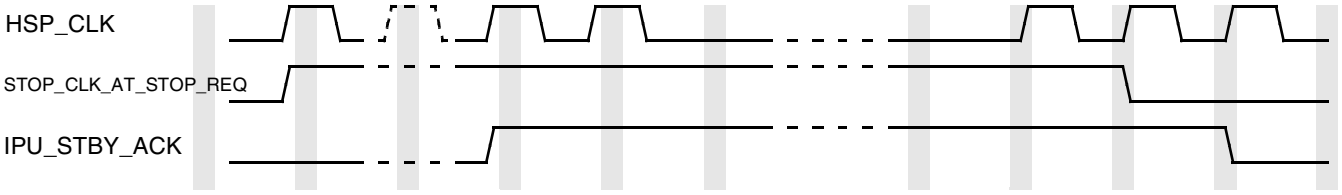


Figure 42-464. Entering and Exiting STOP Mode

Wake up from STOP mode

When the SoC decides to wake up from STOP mode it should

- Resume the clock to the IPUv3EX
- Negate the stop_clk_at_stop_req or stop_clk_at_wait_req signal.
- The IPUv3EX will then negate the ipu_stby_ack signal
- The IPUv3EX will resume screen refresh.

42.3.12.10.2 Power Gating

On power gating mode the clock to the IPUv3EX is stopped and part of the IPUv3EX is powered down. Only the memories that are essential for a quick and smooth wake up are kept powered.

The procedure for entering PG mode is the similar to the procedure for entering STOP mode.

1. If the stop request is asserted and the "gpc_pg_ipu" signal is asserted then we are entering PG mode.
2. The IPUv3EX follows the same process as on STOP mode till getting into IPU_IDLE state.
3. The IPUv3EX saves in the SRM the content of the registers of the modules participating in screen refresh. The modules are:
   - DI0 & DI1
   - DC
   - DP
   - DMFC
   - IDMAC
   - IC
   - CM
4. After saving the registers the IPUv3EX will send the ipu_stby_ack signal to the CCM.

5. The CCM will gate off the clock to the IPUv3EX.
6. The GPC will gate off the power to the IPUv3EX. The memories listed below are needed for quick and smooth wake up, therefore the memories listed below are kept powered.

Table 42-438. Memories that are Powered at PG mode

Sub Module	Instance name	Description
IDMAC	ipu_memories_lut0 ipu_memories_lut1	Look UP table memory
IDMAC	ipu_memories_cpmem	Channel parameter memory
CM shadow	ipu_memories_srm	Shadow registers memory
IC + IRT	ipu_memories_tpm	IC's task's parameters memory
DC	ipu_memories_dc_template	DC's template

Wake up from PG mode

1. The GPC resumes power to the IPUv3EX
2. The GPC drives the signal gpc_ipu_pwgtd indicating that the IPUv3EX wakes up from PG mode.
3. The SoC's reset controller resets the IPUv3EX by asserting the ipg_hard_async_reset_b signal.
4. The CCM resumes the clock to the IPUv3EX.
5. The IPUv3EX reads the registers from the SRM and write the content to the module's registers.
6. The IPUv3EX sends the ipu_wakeup_ack to the CCM indicating that the wake up procedure is complete. This signal is negated when gpc_ipu_pwgtd is negated.
7. The IPUv3EX resumes screen refresh.

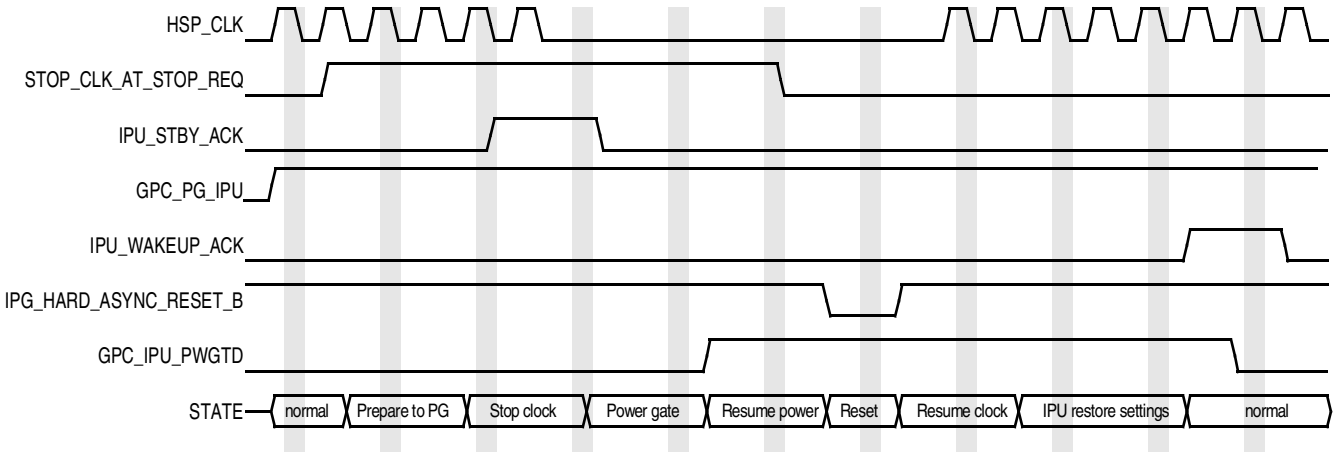


Figure 42-465. Entering and Exiting PG mode

42.3.12.10.3 Low Power Screen Refresh mode - LPSR

On Low Power Screen Refresh mode, the clock to the IPUv3EX is changed to a slower frequency, the IPUv3EX performs only screen refresh to a single display via the DP (channels 23 & 27).

Preparations

1. The user sets the LPSR_MODE bit indicating that the next assertion of stop_clk_at_stop_req OR stop_clk_at_wait_req activates the LPSR procedure.
2. The user moves the IPUv3EX to screen refresh flow. This means that if other tasks are active (flows via CSI, SMFC, or multiple flows to multiple displays) - this tasks needs to be complete and disabled. This is step is fully done by the user (SW). The only flow that remains active is screen refresh to a single display done via the DP (channels 23 & 27)
3. The user stores in the SRM the planned configuration for the modules involved in the LPSR flow. These configuration will e switched with the active registers' settings on later stage. The relevant modules are:
   - DI0 & DI1
   - DC
   - DP
   - DMFC
   - IDMAC
   - CM

Entering LPSR

The flow for entering LPSR is the same as stop mode.

1. The IPUv3EX follows the same procedure as on STOP mode till getting into IPU_IDLE state.
2. IPUv3EX swaps the registers of the relevant modules with the pre stored content from the SRM. The content of the registers of the current flow is stored in the SRM. The IPUv3EX will switch back to this configuration after exiting from LPSR.
3. The IPUv3EX sends the ipu_stby_ack
4. The CCM will gate off the clock to the IPUv3EX.
5. The CCM will switch to the new clock
6. After changing the clock, the CCM asserts ccm_lpsr_ipu, this signal is synched inside the IPUv3EX to the hsp_clk
7. The IPUv3EX will resume screen refresh with the new settings.

Exit from LPSR

1. The CCM negates ccm_lpsr_ipu.
2. The CM sends standby request to the DI
3. The DI should complete processing the current frame and stop the clock to the display and send an acknowledge signal to the CM.

4. The SRM swaps the registers' configuration. current configuration (LPSR) is saved in the SRM, the previous (original) configuration is stored in the modules' registers.
5. IPUv3EX asserts the ipu_wakeup_ack signal indicating that it is now safe to leave LPSR mode.
6. CCM stops the clocks to IPUv3EX
7. CCM resumes the clock to the IPUv3EX - This clock is the same clock used prior to entering the LPSR mode.
8. CCM negates the stop_clk_at_stop_req (or stop_clk_at_wait_req)
9. IPUv3EX negates the ipu_stby_ack and the ipu_wakeup_ack signals
10. The IPUv3EX resumes screen refresh with the original settings.

The diagram below illustrates the procedure for entering and leaving LPSR mode

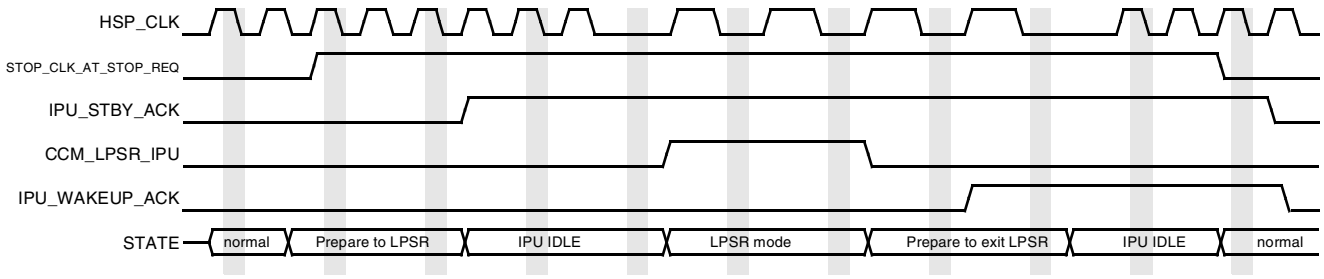


Figure 42-466. Entering and Exiting LPSR mode

42.3.13 IPUv3EX Diagnostics Unit

The IPUv3EX has a 16 bit diagnostic bus called ipu_diagbus. This bus can be connected at SoC level so these 16 outputs of the IPUv3EX can be routed to the SoC's pins and used for signals monitoring during debug.

The internal signals that can be monitored are internal status signals. There are 18 groups of signals. In order to route the selected group the user need to set the IPU_DIAGBUS_MODE and select the correct group. The modules are arranged in groups as described on the tables below.

Table 42-439. IPUv3EX diag bus groups 0-3

	group0	group1	group2	group3
0	IDMAC_NFACK_6	DC_TEARING_ERR_6	IDMAC_EOF_40	IDMAC_EOF_5
1	IDMAC_NFACK_4	DC_TEARING_ERR_2	IDMAC_NFACK_40	IDMAC_NFACK_5
2	CSI1_PUPE	DC_TEARING_ERR_1	DC_ASYNC_STOP	ISP_RAM_HIST_OF

Table 42-439. IPUv3EX diag bus groups 0-3

	group0	group1	group2	group3
3	CSI0_PUPE	DC_FC_6	DMA_CH_CUR_BUF_40	
4		DC_FC_4	DI0_READ_FIFO_FULL	DMA_CH_CUR_BUF_5
5	SMFC3_FRM_LOST	DC_FC_3	DI0_READ_FIFO_EMPTY	DMA_CH_ALT_CUR_BUF_5
6	SMFC2_FRM_LOST	DC_FC_2	DI1_READ_FIFO_FULL	
7	SMFC1_FRM_LOST	DC_FC_1	DI1_READ_FIFO_EMPTY	DMA_CH_ALT_BUF0_RDY_5
8	SMFC0_FRM_LOST	DC_FC_0	DC_TRIPLE_BUF_CNT_EMPTY_1	DMA_CH_ALT_BUF1_RDY_5
9	DMA_CH_ALT_CUR_BUF_52	DC_6_SRM_STAT	DC_TRIPLE_BUF_CNT_FULL_1	DMFC_FIFO_EMPTY_9
10	CSI1_SRM_STAT	DC_2_SRM_STAT	DC_TRIPLE_BUF_DATA_EMPTY_0	DMFC_FIFO_EMPTY_8
11	CSI0_SRM_STAT	DC_ASYNC1_CUR_FLOW	DC_TRIPLE_BUF_DATA_FULL_0	DMFC_FIFO_EMPTY_6
12	IDMAC_EOF_8	Reserved	DC_TRIPLE_BUF_CNT_EMPTY_0	DMFC_FIFO_EMPTY_1
13	IDMAC_EOF_9	Reserved	DC_TRIPLE_BUF_CNT_FULL_0	DMFC_FIFO_FULL_9
14	IDMAC_EOF_10	DC_TRIPLE_BUF_DATA_EMPTY_1	DMFC_FIFO_EMPTY_4	DMFC_FIFO_FULL_8
15	IDMAC_EOF_13	DC_TRIPLE_BUF_DATA_FULL_1	DMFC_FIFO_FULL_4	DMFC_FIFO_FULL_6

Table 42-440. IPUv3EX diag bus groups 4-7

	group4	group5	group6	group7
0	DI0_CNT_EN_PRE_10	IDMAC_EOF_52	IDMAC_EOF_28	IDMAC_EOF_31
1	DI0_CNT_EN_PRE_9	IDMAC_NFACK_52	IDMAC_EOF_27	IDMAC_EOF_51
2	DI0_CNT_EN_PRE_8	DI1_SYNC_DISP_ERR	IDMAC_EOF_24	IDMAC_NFACK_31
3	DI0_CNT_EN_PRE_6	DI1_CNT_EN_PRE_8	IDMAC_EOF_23	IDMAC_NFACK_28
4	DI0_CNT_EN_PRE_5	DI1_CNT_EN_PRE_3	DC_DP_START	IDMAC_NFACK_27
5	DI0_CNT_EN_PRE_4	DI1_DISP_CLK_EN_PRE	DP_ASF_BRAKE	IDMAC_NFACK_24
6	DI0_CNT_EN_PRE_3	DI_VSYNC_PRE_1	DP_SF_BRAKE	IDMAC_NFACK_23
7	DI0_CNT_EN_PRE_2	DMA_CH_CUR_BUF_52	DP_ASF_END	IDMAC_NFACK_51
8	DI0_CNT_EN_PRE_1	DI1_SRM_STAT	DP_ASF_START	DMA_CH_CUR_BUF_31
9	DI0_DISP_CLK_EN_PRE	DMA_CH_ALT_BUF0_RDY_52	DP_SF_END	DMA_CH_CUR_BUF_28
10	DI_VSYNC_PRE_0	DMA_CH_ALT_BUF1_RDY_52	DP_SF_START	DMA_CH_CUR_BUF_27
11	DMA_CH_ALT_CUR_BUF_24	DI1_CNTR_FIFO_FULL	DP_A1_SRM_STAT	DMA_CH_CUR_BUF_24
12	DI0_SRM_STAT	DI1_CNTR_FIFO_EMPTY	DP_A0_SRM_STAT	DMA_CH_CUR_BUF_23
13	DMA_CH_ALT_BUF0_RDY_24	DMFC_FIFO_EMPTY_5	DP_S_SRM_STAT	DMA_CH_CUR_BUF_51

Table 42-440. IPUv3EX diag bus groups 4-7

	group4	group5	group6	group7
14	DI0_CNTR_FIFO_FULL	DMFC_FIFO_EMPTY_0	DP_ASYNC_CUR_FLOW	DMA_CH_CUR_BUF_47
15	DI0_CNTR_FIFO_EMPTY	DMFC_FIFO_FULL_5	Reserved	DMA_CH_ALT_BUF1_RDY_24

Table 42-441. IPUv3EX diag bus groups 8-11

	group8	group9	group10	group11
0	IDMAC_EOF_3	IDMAC_EOF_29	IDMAC_EOF_18	IDMAC_EOF_15
1	IDMAC_EOF_2	IDMAC_EOF_6	IDMAC_EOF_17	IDMAC_EOF_14
2	IDMAC_EOF_1	IDMAC_EOF_4	IDMAC_EOF_7	IDMAC_EOF_12
3	IDMAC_EOF_0	IDMAC_NFACK_29	IDMAC_NFACK_18	IDMAC_EOF_11
4	IDMAC_NFACK_3	DMA_CH_CUR_BUF_29	IDMAC_NFACK_17	IDMAC_NFACK_15
5	IDMAC_NFACK_2	DMA_CH_CUR_BUF_6	IDMAC_NFACK_7	IDMAC_NFACK_14
6	IDMAC_NFACK_1	DMA_CH_CUR_BUF_4	DI0_CNT_EN_PRE_7	IDMAC_NFACK_12
7	IDMAC_NFACK_0	DMA_CH_ALT_CUR_BUF_29	DMA_CH_CUR_BUF_18	IDMAC_NFACK_11
8	DMA_CH_CUR_BUF_3	DMA_CH_ALT_CUR_BUF_6	DMA_CH_CUR_BUF_17	DMA_CH_CUR_BUF_15
9	DMA_CH_CUR_BUF_2	DMA_CH_ALT_CUR_BUF_4	DMA_CH_CUR_BUF_7	DMA_CH_CUR_BUF_14
10	DMA_CH_CUR_BUF_1	DMA_CH_ALT_BUF0_RDY_29	DMA_CH_ALT_CUR_BUF_7	DMA_CH_CUR_BUF_12
11	DMA_CH_CUR_BUF_0	DMA_CH_ALT_BUF0_RDY_6	DMA_CH_ALT_BUF0_RDY_7	DMA_CH_CUR_BUF_11
12	DMFC_FIFO_EMPTY_3	DMA_CH_ALT_BUF0_RDY_4	DMA_CH_ALT_BUF1_RDY_7	DMFC_FIFO_EMPTY_11
13	DMFC_FIFO_EMPTY_2	DMA_CH_ALT_BUF1_RDY_29	DMFC_FIFO_EMPTY_7	DMFC_FIFO_EMPTY_10
14	DMFC_FIFO_FULL_3	DMA_CH_ALT_BUF1_RDY_6	DMFC_FIFO_FULL_7	DMFC_FIFO_FULL_11
15	DMFC_FIFO_FULL_2	DMA_CH_ALT_BUF1_RDY_4	DMFC_FIFO_FULL_0	DMFC_FIFO_FULL_10

Table 42-442. IPUv3EX diag bus groups 12-15

	group12	group13	group14	group15
0	IDMAC_EOF_50	IDMAC_EOF_44	IDMAC_EOF_22	IC_VF_BUF_OVF
1	IDMAC_EOF_49	IDMAC_EOF_43	IDMAC_EOF_21	IC_ENC_BUF_OVF
2	IDMAC_EOF_48	IDMAC_EOF_42	IDMAC_EOF_20	IC_BAYER_BUF_OVF
3	IDMAC_EOF_47	IDMAC_EOF_41	IDMAC_EOF_33	IC_BAYER_FRM_LOST_ERR
4	IDMAC_EOF_46	IDMAC_NFACK_44	IDMAC_NFACK_22	IC_ENC_FRM_LOST_ERR
5	IDMAC_EOF_45	IDMAC_NFACK_43	IDMAC_NFACK_21	IC_VF_FRM_LOST_ERR
6	IDMAC_NFACK_50	IDMAC_NFACK_42	IDMAC_NFACK_20	DMA_CH_CUR_BUF_50
7	IDMAC_NFACK_49	IDMAC_NFACK_41	IDMAC_NFACK_33	IDMAC_NFACK_8
8	IDMAC_NFACK_48	DMA_CH_CUR_BUF_44	DI0_SYNC_DISP_ERR	IDMAC_NFACK_9
9	IDMAC_NFACK_47	DMA_CH_CUR_BUF_43	DMA_CH_CUR_BUF_22	IDMAC_NFACK_10

Table 42-442. IPUv3EX diag bus groups 12-15

	group12	group13	group14	group15
10	IDMAC_NFACK_46	DMA_CH_CUR_BUF_42	DMA_CH_CUR_BUF_21	IDMAC_NFACK_13
11	IDMAC_NFACK_45	DMA_CH_CUR_BUF_41	DMA_CH_CUR_BUF_20	Reserved
12	DMA_CH_CUR_BUF_49	DMA_CH_ALT_CUR_BUF_41	DMA_CH_CUR_BUF_33	Reserved
13	DMA_CH_CUR_BUF_48	DMA_CH_ALT_BUF0_RDY_41	DMA_CH_ALT_CUR_BUF_33	VDI_FIFO1_OVF
14	DMA_CH_CUR_BUF_46	DMA_CH_ALT_BUF1_RDY_41	DMA_CH_ALT_BUF0_RDY_33	DMFC_IC_BUFFER_EMPTY
15	DMA_CH_CUR_BUF_45	DMFC_FIFO_FULL_1	DMA_CH_ALT_BUF1_RDY_33	DMFC_IC_BUFFER_FULL

Table 42-443. IPUv3EX diag bus group 16

	group16
0	CSI2MEM_SMFC3_TSTAT
1	
2	CSI2MEM_SMFC2_TSTAT
3	
4	CSI2MEM_SMFC1_TSTAT
5	
6	CSI2MEM_SMFC0_TSTAT
7	
8	DC_ASYNC1_TSTAT
9	
10	
11	DC_ASYNC0_TSTAT
12	
13	DP_ASYNC_TSTAT
14	
15	

Table 42-444. IPUv3EX diag bus group 17

	group17
0	MEM2PRP_TSTAT
1	
2	
3	PP_ROT_TSTAT
4	

Table 42-444. IPUv3EX diag bus group 17 (continued)

	group17
5	VF_ROT_TSTAT
6	
7	ENC_ROT_TSTAT
8	
9	PP_TSTAT
10	
11	VF_TSTAT
12	
13	ENC_TSTAT
14	
15	Reserved

Chapter 43
Keypad Port (KPP)

43.1 Introduction

The Keypad Port (KPP) is a 16-bit peripheral that can be used as a keypad matrix interface or as general purpose input/output (I/O). Figure 43-1 shows the KPP block diagram.

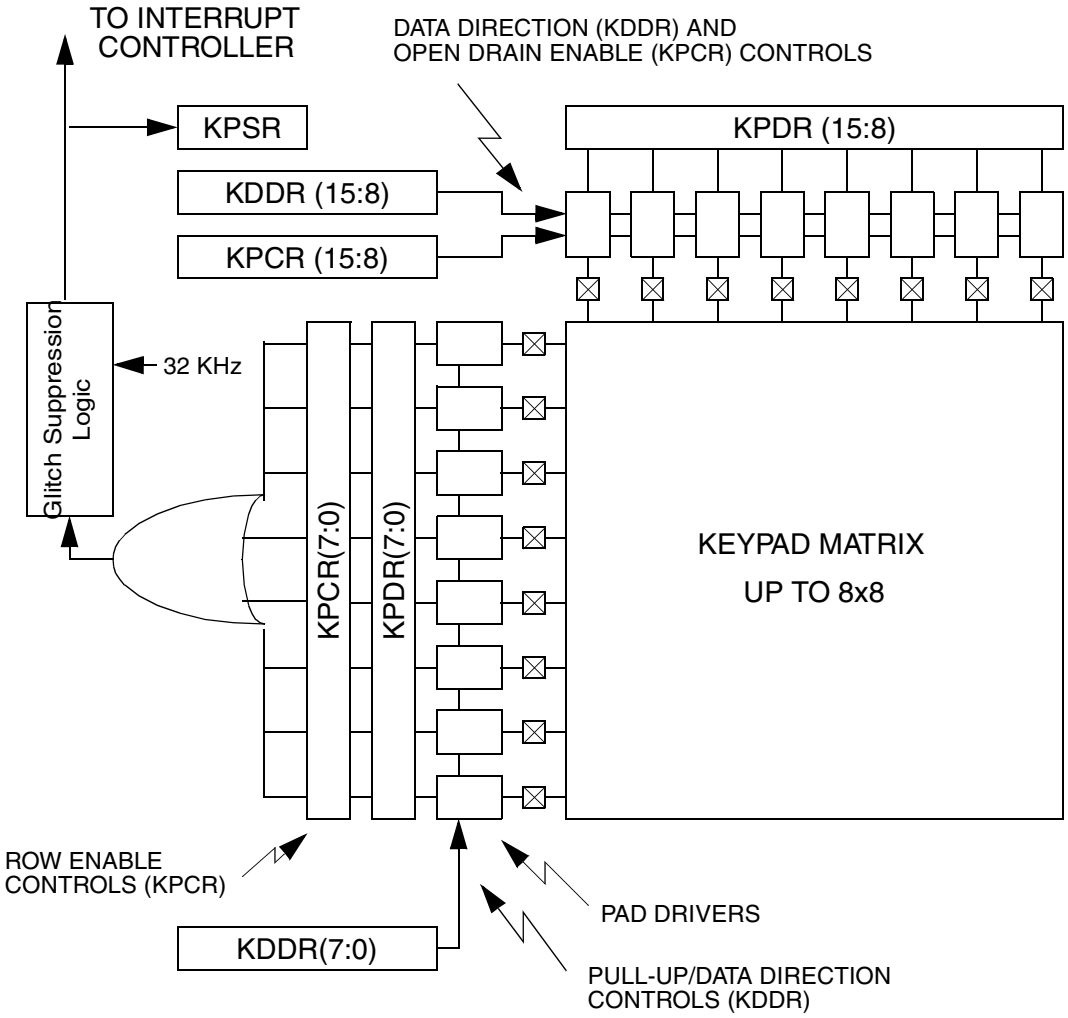


Figure 43-1. KPP Peripheral Block Diagram

43.2 Overview

The KPP is designed to interface with the keypad matrix with 2-point contact or 3-point contact keys. The KPP is designed to simplify the software task of scanning a keypad matrix. With appropriate software support, the KPP is capable of detecting, debouncing, and decoding one or multiple keys pressed simultaneously on the keypad.

43.2.1 Features

The KPP includes these distinctive features:

- Supports up to an 8×8 external key pad matrix
- Port pins can be used as general purpose I/O
- Open drain design
- Glitch suppression circuit design
- Multiple-key detection
- Long key-press detection
- Standby key-press detection
- Synchronizer chain clear
- Supports a 2-point and 3-point contact key matrix

43.2.2 Modes of Operation

This module supports the following modes of operation:

- Run Mode—This is the normal functional mode in which the KPP can detect any key press event.
- Low Power Modes—The keypad can detect any key press even in low power modes (when there is no MCU clock).

43.3 External Signal Description

There are 16 pins dedicated to the KPP. Keypads of any configuration up to eight rows and eight columns are supported through the software configuration of the peripheral pins. Any pins not used for the keypad are available as general purpose I/O. The registers are configured such that the pins can be treated as an I/O port up to 16 bits wide.

See Table 43-1 for the list of external signals.

Table 43-1. Signal Properties

Name	Port	Function	Reset State	Pull up
ipp_ind_col[7:0]	—	Column input pin, from chip	0	Active[1]
ipp_ind_row[7:0]	—	Row input pin, from chip	1	Active[1]
ipp_do_col[7:0]	—	Column output pin going to chip	0	—
ipp_obe_col[7:0]	—	Enables the corresponding column outputs	0	—

Table 43-1. Signal Properties (continued)

Name	Port	Function	Reset State	Pull up
ipp_ode_col[7:0]	—	Used to set the column outputs	0	—
ipp_do_row[7:0]	—	Row output pin going to chip	0	—
ipp_obe_row[7:0]	—	Enables the corresponding row output	0	—

[1] The corresponding pads are required to be pull-up enabled.

43.3.1 Input Pins

Any of the 16 pins associated with the KPP can be configured as inputs by writing a “0” to the appropriate bits in the KDDR. Additionally, the least significant 8 bits (ROW inputs) corresponding to KDDR7:0 have internal pull-ups, which are enabled when the pin is used as an input.

43.3.2 Output Pins

Any of the 16 pins associated with the KPP can be configured as outputs by writing the appropriate bits in the KDDR to a “1”. Additionally, the 8 most significant bits (15–8) can be designated as open drain outputs by writing a “1” to the appropriate bits in the KPCR. The lower 8 bits (7–0) are always in “totem pole” style, driven when configured as outputs. See Table 43-2.

Table 43-2. Keypad Port Column Modes

KDDR (15:8)	KPCR (15:8)	Pin Function
0	x	Input
1	0	Totem-Pole Output
1	1	Open-Drain Output

NOTE

Totem pole capability should be provided for column pins. Totem pole configuration helps for a faster discharge of keypad capacitance when all columns need to be quickly brought to a “1” during the scan routine. With this configuration, a time delay between the scanning of two subsequent columns is reduced.

43.4 Memory Map and Register Definition

The KPP module contains four registers. Section 43.4.3, Register Descriptions” provides detailed descriptions of the KPP registers.

43.4.1 KPP Memory Map

Table 43-3 shows the KPP memory map.

Table 43-3. KPP Memory Map

Address	Use	Access	Reset Value	Section/Page
0xBASE+0x000 (KPCR)	Keypad Control Register	R/W	0x0000	43.4.3.1/43-5
0xBASE+0x002 (KPSR)	Keypad Status Register	R/W	0x0000	43.4.3.2/43-6
0xBASE+0x004 (KDDR)	Keypad Data Direction Register	R/W	0x0000	43.4.3.3/43-7
0xBASE+0x006 (KPDR)	Keypad Data Register	R/W	0xXXXX	43.4.3.4/43-8

43.4.2 Register Summary

Figure 43-2 shows the key to the register fields and Table 43-4 shows the register figure conventions.

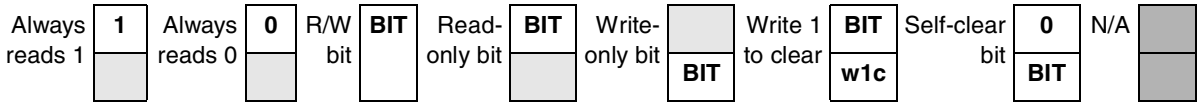


Figure 43-2. Key to Register Fields

Table 43-4. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 43-5 shows the KPP register summary.

Table 43-5. KPP Register Summary

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (KPCR)	R	KCO7	KCO6	KCO5	KCO4	KCO3	KCO2	KCO1	KCO0	KRE7	KRE6	KRE5	KRE4	KRE3	KRE2	KRE1	KRE0
	W																
0xBASE+0x002 (KPSR)	R	0	0	0	0	0	0	KRIE	KDIE	0	0	0	0	0	0	KPKR	KPKD
	W													KRSS	KDSC	w1c	w1c
0xBASE+0x004 (KDDR)	R	KCD D7	KCD D6	KCD D5	KCD D4	KCD D3	KCD D2	KCD D1	KCD D0	KRD D7	KRD D6	KRD D5	KRD D4	KRD D3	KRD D2	KRD D1	KRD D0
	W																
0xBASE+0x006 (KPDR)	R	KCD7	KCD6	KCD5	KCD4	KCD3	KCD2	KCD1	KCD0	KRD7	KRD6	KRD5	KRD4	KRD3	KRD2	KRD1	KRD0
	W																

43.4.3 Register Descriptions

This section consists of register descriptions. Each register is listed in the order of its address.

43.4.3.1 Keypad Control Register (KPCR)

The Keypad Control Register determines which of the eight possible column strobes are to be open drain when configured as outputs, and which of the eight row sense lines are considered in generating an interrupt to the core.

It is up to the programmer to ensure that pins being used for functions other than the keypad are properly disabled. The KPCR register is byte- or half-word–addressable.

Figure 43-3 shows the valid bits in the KPCR register, and Table 43-6 provides its field descriptions.

0xBASE+0x000 (KPCR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	KCO7	KCO6	KCO5	KCO4	KCO3	KCO2	KCO1	KCO0	KRE7	KRE6	KRE5	KRE4	KRE3	KRE2	KRE1	KRE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 43-3. KPCR Register

Table 43-6. Keypad Control Register Field Descriptions

Field	Description
15–8 KCO	Keypad Column Strobe Open-Drain Enable. Setting a column open-drain enable bit (KCO7–KCO0) disables the pull-up driver on that pin. Clearing the bit allows the pin to drive to the high state. This bit has no effect when the pin is configured as an input. 0 Column strobe output is totem pole drive. 1 Column strobe output is open drain. **Note:** Configuration of external port control logic (for example, GPIO) should be done properly so that the KPP module controls an open-drain enable of the pin.
7–0 KRE	Keypad Row Enable. Setting a row enable control bit in this register enables the corresponding row line to participate in interrupt generation. Likewise, clearing a bit disables that row from being used to generate an interrupt. This register is cleared by a reset, disabling all rows. The row-enable logic is independent of the programmed direction of the pin. Writing a “0” to the data register of the pins configured as outputs will cause a keypad interrupt to be generated if the row enable associated with that bit is set. 0 Row is not included in the keypad key press detect. 1 Row is included in the keypad key press detect.

43.4.3.2 Keypad Status Register (KPSR)

The Keypad Status Register reflects the state of the key press detect circuit. The KPSR register is byte- or half-word addressable.

Figure shows the KPSR register, and Table 43-7 provides its field descriptions.

0xBASE+0x002 (KPSR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	KRIE	KDIE	0	0	0	0	0	0	KPKR	KPKD
W													KRSS	KDSC	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 43-4. KPSR Register Diagram

Table 43-7. Keypad Status Register Field Descriptions

Field	Description
15–10	Reserved
9 KRIE	Keypad Release Interrupt Enable. The software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain. 0 No interrupt request is generated when KPKR is set. 1 An interrupt request is generated when KPKR is set.

Table 43-7. Keypad Status Register Field Descriptions (continued)

Field	Description
8 KDIE	Keypad Key Depress Interrupt Enable. Software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice-versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain. 0 No interrupt request is generated when KPKD is set. 1 An interrupt request is generated when KPKD is set.
7–4	Reserved, should be cleared
3 KRSS	Key Release Synchronizer Set. Self-clear bit. The Key release synchronizer is set by writing a logic one into this bit. Reads return a value of “0”. 0 No effect 1 Set bits which sets keypad release synchronizer chain
2 KDSC	Key Depress Synchronizer Clear. Self-clear bit. The Key depress synchronizer is cleared by writing a logic “1” into this bit. Reads return a value of “0”. 0 No effect 1 Set bits that clear the keypad depress synchronizer chain
1 KPKR	Keypad Key Release. The keypad key release (KPKR) status bit is set when all enabled rows are detected high after synchronization (the KPKR status bit will be set when cleared by a reset). The KPKR bit may be used to generate a maskable key release interrupt. The key release synchronizer may be set high by software after scanning the keypad to ensure a known state. Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by software prior to the system exiting the state it represents. 0 No key release detected 1 All keys have been released Reset value of register is “0” as long as reset is asserted. However when reset is de-asserted, the value of the register depends upon the external row pins and can become “1”.
0 KPKD	Keypad Key Depress. The keypad key depress (KPKD) status bit is set when one or more enabled rows are detected low after synchronization. The KPKD status bit remains set until cleared by the software. The KPKD bit may be used to generate a maskable key depress interrupt. If desired, the software may clear the key press synchronizer chain to allow a repeated interrupt to be generated while a key remains pressed. In this case, a new interrupt will be generated after the synchronizer delay (4 cycles of the 32 KHz clock) elapses if a key remains pressed. This functionality can be used to detect a long key press. This allows detection of additional key presses of the same key or other keys. Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by the software prior to the system exiting the state it represents. 0 No key presses detected 1 A key has been depressed

43.4.3.3 Keypad Data Direction Register (KDDR)

The bits in the KDDR control the direction of the keypad port pins. The upper eight bits in the register affect the pins designated as column strobes, while the lower eight bits affect the row sense pins. Setting any bit in this register configures the corresponding pin as an output. Clearing any bit in this register configures the corresponding port pin as an input. For the Keypad Row DDR, an internal pull-up is enabled

if the corresponding bit is clear. This register is cleared by a reset, configuring all pins as inputs. The KDDR register is byte- or half-word addressable.

NOTE

When a pin is used as row pin for keypad purposes, all corresponding pull-ups should be enabled at the upper level (for example, IOMUX) when the bit in row DDR is cleared.

Figure 43-5 shows the valid bits in the KDDR register, and Table 43-8 provides its field descriptions.

0xBASE+0x004 (KDDR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	KCDD 7	KCDD 6	KCDD 5	KCDD 4	KCDD 3	KCDD 2	KCDD 1	KCDD 0	KRDD 7	KRDD 6	KRDD 5	KRDD 4	KRDD 3	KRDD 2	KRDD 1	KRD D0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 43-5. KDDR Register Diagram

Table 43-8. Keypad Data Direction Register Field Descriptions

Field	Description
15–8 KCDD	Keypad Column Data Direction Register. Setting any bit configures the corresponding pin as an output. 0 COLn pin is configured as an input. 1 COLn[1] pin is configured as an output.
7–0 KRDD	Keypad Row Data Direction. Setting any bit configures the corresponding pin as an output. 0 ROWn pin configured as an input. 1 ROWn[2] pin configured as an output.

[1] n=7-0
[2] n=7-0

43.4.3.4 Keypad Data Register (KPDR)

This 16-bit register is used to access the column and row data. Data written to this register is stored in an internal latch, and for each pin configured as an output, the stored data is driven onto the pin. A read of this register returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register.

The KPDR register is byte- or half-word–addressable. This register is not initialized by a reset. Valid data should be written to this register before any bits are configured as outputs.

Figure 43-6 shows the KPDR register, and Table 43-9 provides its field descriptions.

0xBASE+0x006 (KPDR) Access: User read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	KCD7	KCD6	KCD5	KCD4	KCD3	KCD2	KCD1	KCD0	KRD7	KRD6	KRD5	KRD4	KRD3	KRD2	KRD1	KRD 0
Reset	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Figure 43-6. KPDR Register Diagram

Table 43-9. Keypad Data Register Field Descriptions

Field	Description
15-8 KCD[	Keypad Column Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write “0” from/to column ports 1 Read/Write “1” from/to column ports
7-0 KRD	Keypad Row Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write “0” from/to row ports 1 Read/Write “1” from/to row ports

43.5 Functional Description

The Keypad Port (KPP) is designed to simplify the software task of scanning a keypad matrix. With appropriate software support and matrix organization, the KPP is capable of detecting, debouncing, and decoding one or more keys pressed simultaneously on the keypad.

Logic in the KPP is capable of detecting a key press even while the processor is in one of the low power standby modes providing that a 32 KHz clock is on. The KPP may generate a CPU interrupt any time a key press or key release is detected. This interrupt is capable of forcing the processor out of a low power mode.

43.5.1 Keypad Matrix Construction

The KPP is designed to interface to a keypad matrix, which shorts the intersecting row and column lines together whenever a key is depressed. The interface is not optimized for any other switch configuration.

43.5.2 Keypad Port Configuration

The software must initialize the KPP for the size of the keypad matrix. Pins connected to the keypad columns should be configured as open-drain outputs. Pins connected to the keypad rows should be configured as inputs. On-chip, pull-up resistors should be implemented for active keypad rows.

In addition to enabled row inputs in the Keypad Control register, corresponding interrupt (depress or/and release) must also be enabled to generate an interrupt.

Discrete switches that are not part of the matrix may be connected to any unused row inputs. The second terminal of the discrete switch is connected to ground. The hardware detects closures of these switches without the need for software polling.

43.5.3 Keypad Matrix Scanning

Keypad scanning is performed by a software loop that walks a zero across each of the keypad columns, reading the value on the rows at each step. The process is repeated several times in succession, with the results of each pass optionally compared to those from the previous pass. When several (3 or 4) consecutive scans yield the same key closures, a valid key press has been detected. Software then can decode exactly which switch was depressed and pass the value up to the next higher software layer.

The basic debouncing period, which must be defined in the software routine, may be controlled with an internal timer. The basic period is the period between the scan of two consecutive columns, so the debouncing time between two consecutive scans of the whole matrix shall be the number of columns multiplied by the basic period.

43.5.4 Keypad Standby

There is no need for the CPU to continually scan the keypad. Between key presses, the keypad can be left in a state that requires no software intervention until the next key press is detected. To place the keypad in a standby state, software should write all column outputs low. Row inputs are left enabled. At this point, the CPU can attend to other tasks or revert to a low power standby mode. The KPP will interrupt the CPU if any key is pressed.

Upon receiving a keypad interrupt, the CPU should set all the column strobes high, and begin a normal keypad scanning routine to determine which key was pressed. It is important that open-drain drivers be used when scanning to prevent a possible DC path between power and ground through two or more switches.

43.5.5 Glitch Suppression on Keypad Inputs

A glitch suppression circuit qualifies the keypad inputs to prevent noise from inadvertently interrupting the CPU. The circuit is a 4-state synchronizer clocked from a 32 KHz clock source. This clock must continue to run in any low power mode where the keypad is a wake-up source, as the CPU interrupt is generated from the synchronized input. An interrupt is not generated until all four synchronizer stages have latched a valid key assertion. This guarantees the filtering out of any noise less than three clock periods (for 32 KHz clock: 93.75 µs) in duration. Noise filtering of the duration between three to four clock periods (for the 32 KHz clock: between 93.75 µs and 125 µs) cannot be guaranteed. The interrupt output is latched in an S-R latch and remains asserted until cleared by the software. The Set input of the latch is rising-edge clocked. See Figure 43-7.

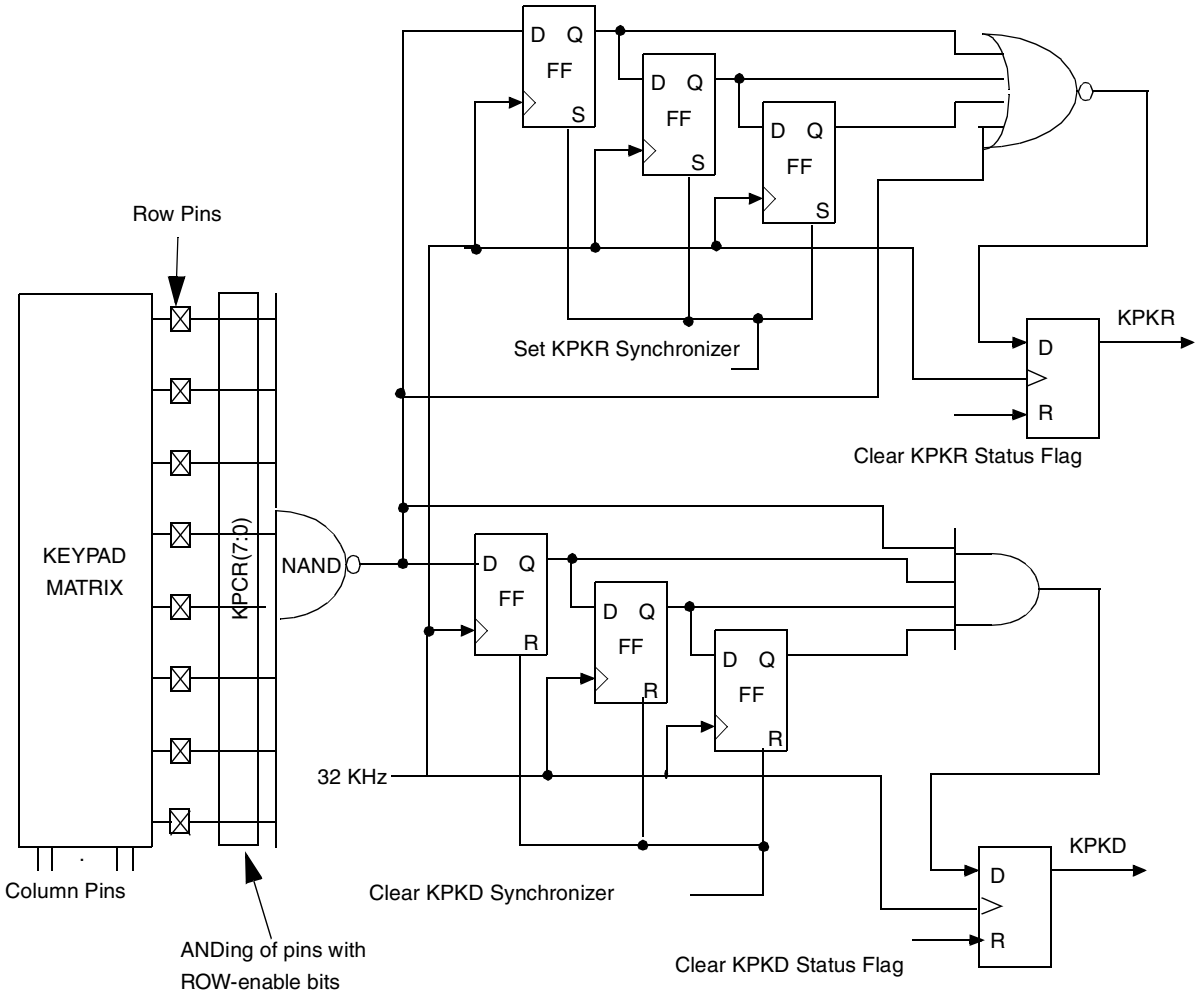


Figure 43-7. Keypad Synchronizer Functional Diagram

43.5.6 Multiple Key Closures

Using the key press and Key release interrupts, the software can detect multiple keys or achieve n key rollover. The key scanning routine can be programmed accordingly. Refer to Section 43.6, Initialization/Application Information," for more information.

See Figure 43-6 and Figure 43-9 for illustrations of the interfacing of a 2-contact keypad matrix with the KPP controller. With proper enabling of row lines and the performing scan-routine, multiple key presses can be detected. When keys present on the same row are pressed, corresponding row lines (multiple lines) become low when the column is driven low during a scan-routine. By reading the data-register, pressed

keys can be detected. Similarly, when keys present on same row line are pressed, the corresponding row line (only one line) becomes low when logic “0” is driven on the column line during a scan-routine.

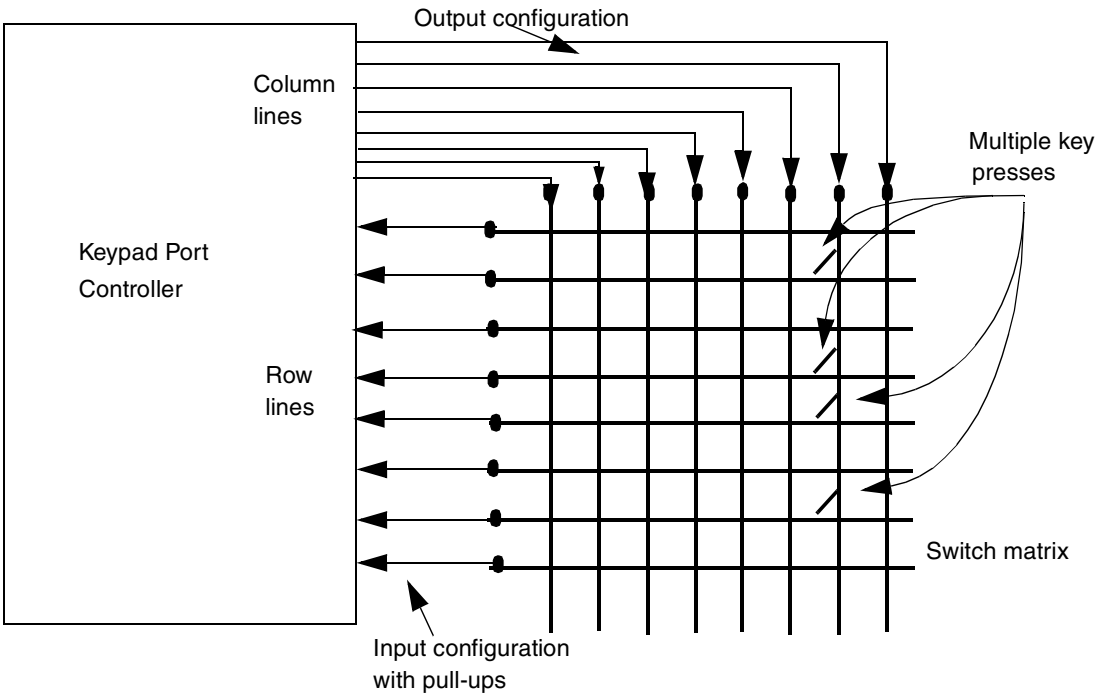


Figure 43-8. Multiple Key Presses on Same Column Line (Simplified View)

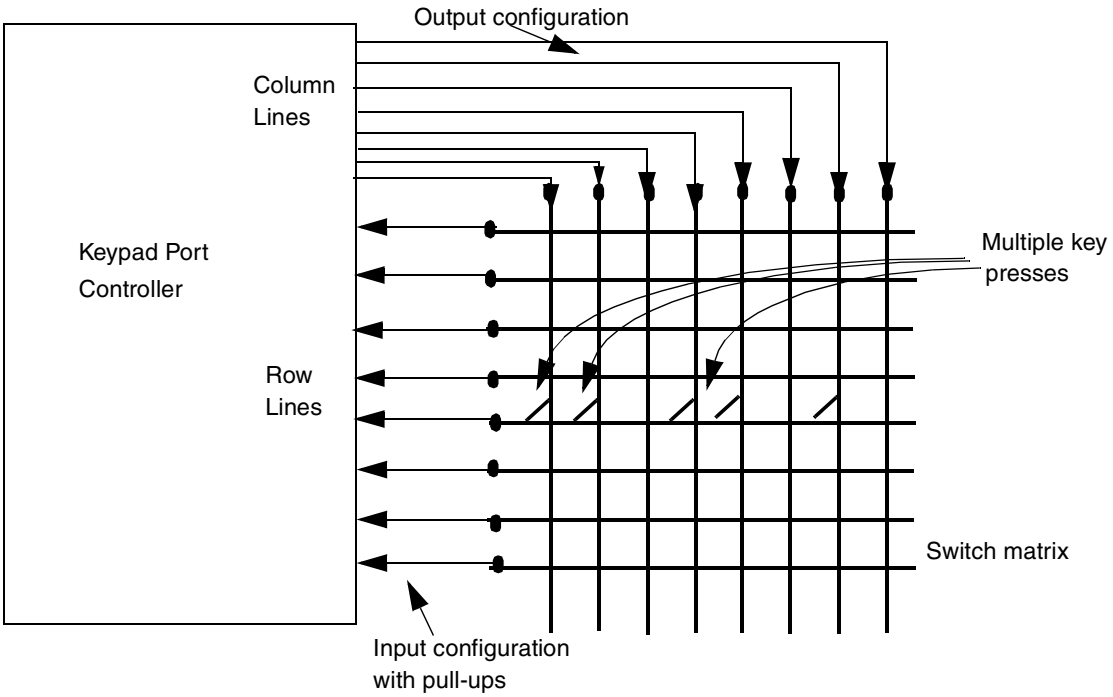


Figure 43-9. Multiple Key Presses on Same Row Line (Simplified View)

NOTE

An n key rollover is a technique with which the system can recognize the order in which keys are pressed.

43.5.6.1 Ghost Key Problem and Correction

The KPP module detects if one or multiple keys are pressed or released. In the case where a simple keypad matrix with two-contact switches is used, there is a chance of “ghost” key detection when three or more keys are pressed. This is a limitation imposed by such a keypad matrix. As can be seen in Figure 43-10, three keys pressed simultaneously can cause a short between the column currently “scanned” by the software and another column. Depending on the location of the third key pressed, a “ghost” key press may be detected.

However, this can be corrected by using a keypad matrix that provides “ghost” key protection. Such a matrix implements a one-way “diode” at all keypad points between rows and columns. This way, the multiple pressing of three keys will not cause a short at a fourth key (see Figure 43-11).

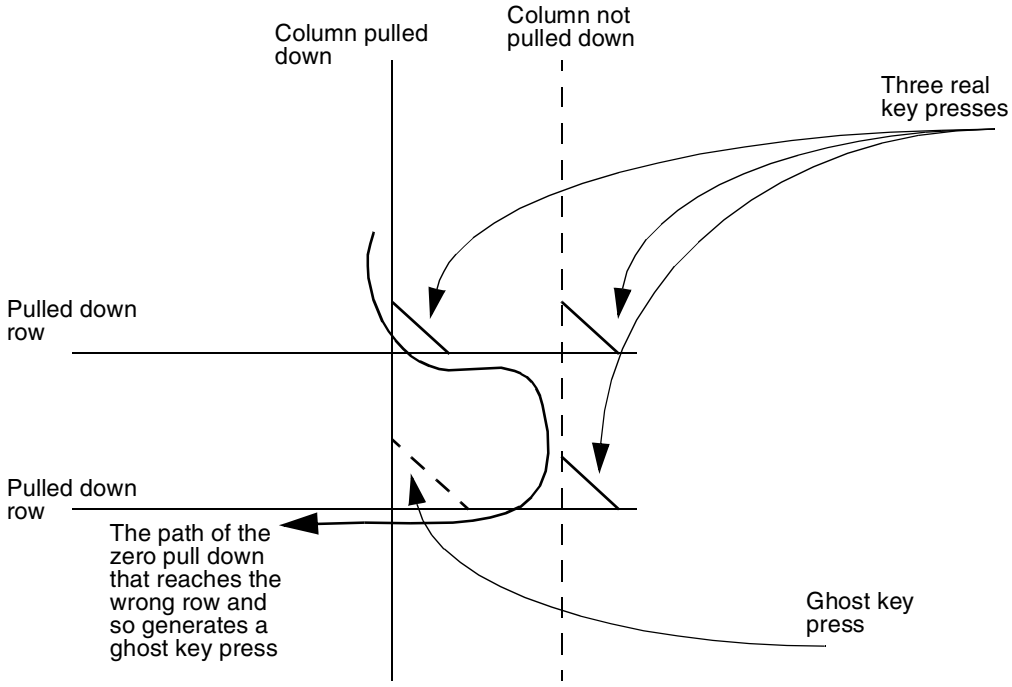


Figure 43-10. Decoding Wrong Three- Key-Presses

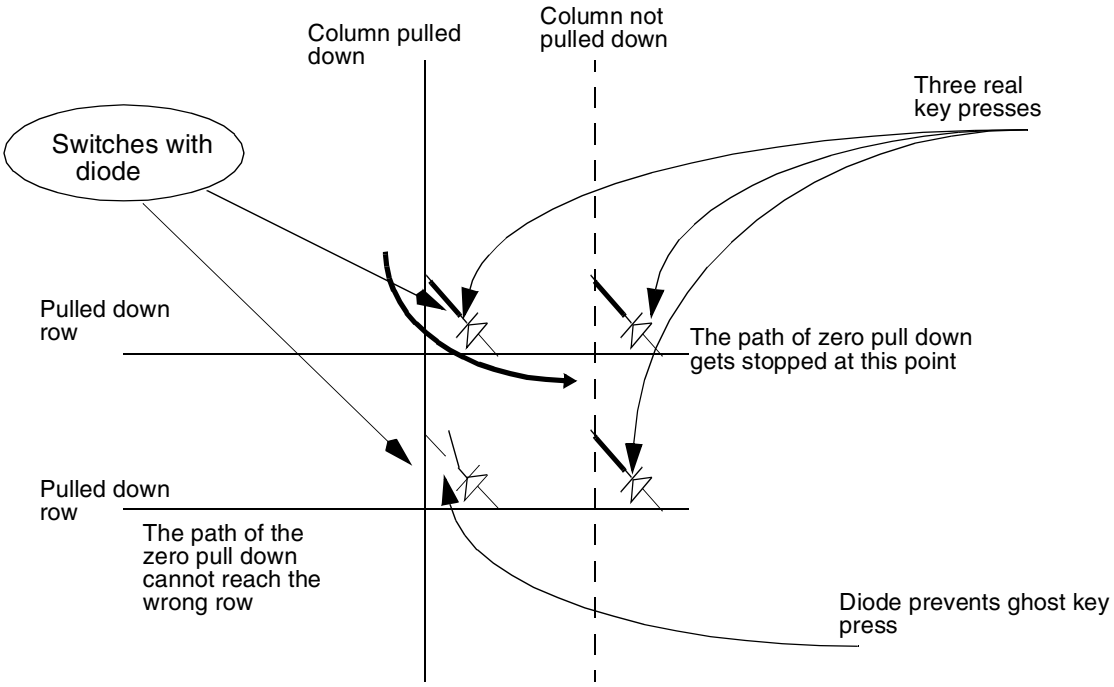


Figure 43-11. Matrix with "Ghost" Key Protections

43.5.7 3-Point Contact Keys Support

The KPP module supports interfacing to a matrix consisting of 3-point contact keys. As shown in Figure 43-12, two points of such a key are connected to keypad lines, while a third point is connected to ground (low logic). The keypad lines should be configured as input and a pull-up should to be present on these lines. When such a key is pressed, corresponding keypad lines go low and an interrupt is generated. There is no need to perform a scanning routine for identification of pressed key as it can be done by reading

the keypad data-register. A limitation with such a matrix is that for every key at least one keypad row line should be used.

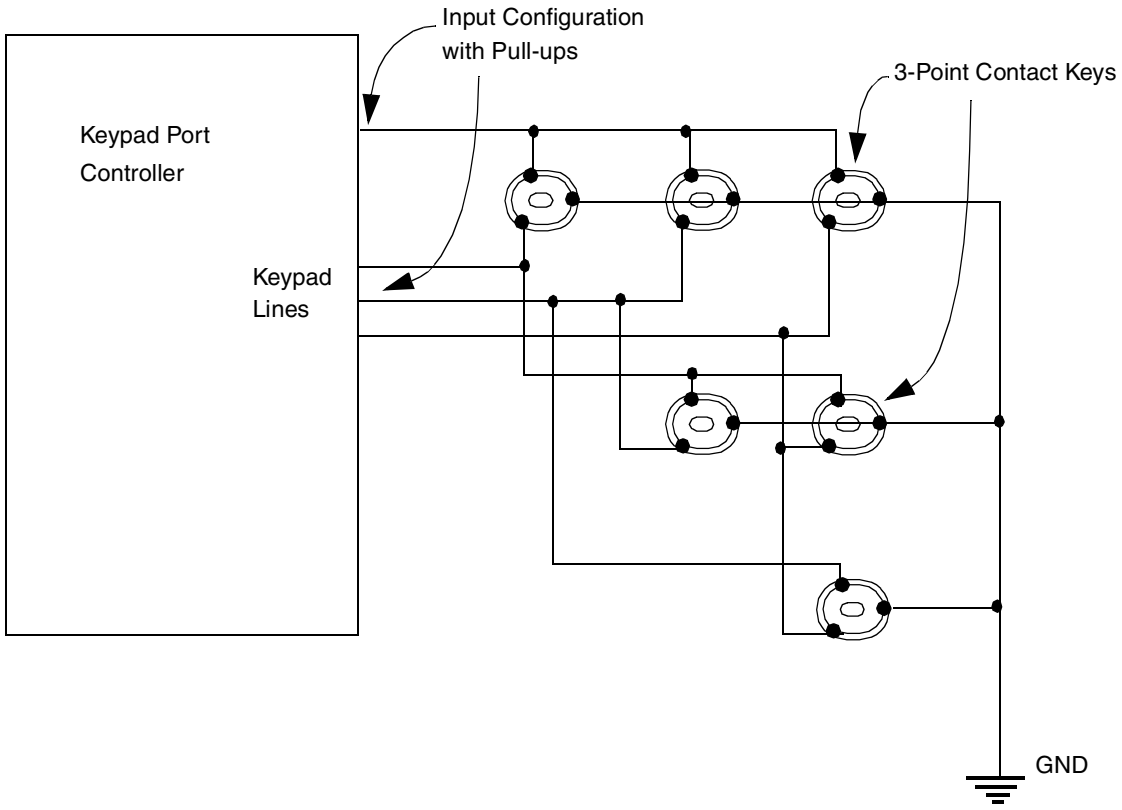


Figure 43-12. KPP Interface with 3-point Contact Key Matrix (Simplified View)

43.6 Initialization/Application Information

This section provides the sequences for typical keypad configuration and scanning and key press interrupt scanning.

43.6.1 Typical Keypad Configuration and Scanning Sequence

Perform the following steps to configure the keypad:

1. Enable the number of rows in the keypad (KPCR[7–0]).
2. Write 0s to KPDR[15–8].
3. Configure the keypad columns as open-drain (KPCR[15–8]).
4. Configure columns as output and rows as input (KDDR[15–0]).
5. Clear the KPKD Status Flag and Synchronizer chain.
6. Set the KDIE control bit, and clear the KRIE control bit (avoid false release events).
7. The system is now in standby mode, awaiting a key press.

43.6.2 Key Press Interrupt Scanning Sequence

Perform the following steps to perform a keypad scanning routine:

1. Disable both (depress and release) keypad interrupts.
2. Write 1s to KPDR[15:8], setting column data to 1s.
3. Configure columns as totem pole outputs (for quick discharging of keypad capacitance).
4. Configure columns as open-drain.
5. Write a single column to 0, and other columns to 1.
6. Sample row inputs and save data. Multiple key presses can be detected on a single column.
7. Repeat Steps 2–6 for remaining columns.
8. Return all columns to 0 in preparation for standby mode.
9. Clear KPKD and KPKR status bit(s) by writing to a “1”; set the KPKR synchronizer chain by writing a “1” to the KRSS register; and clear the KPKD synchronizer chain by writing a “1” to the KDSC register.
10. Re-enable the appropriate keypad interrupt(s) so that the KDIE detects a key hold condition, or the KRIE detects a key-release event.

43.6.3 Additional Comments

The order of key press detection can be done in software only. Therefore, the software may need to run the scan routines at very short intervals of time per the application’s demands. The reason that such functionality cannot be put in the KPP module is that the module is limited by the number of external pins.

For the keys that require a very precise order (such as game keys), individual GPIO pins may be more useful.

Chapter 44
Multi Master Multi Memory Interface (i.MX51)

44.1 Introduction

The Multi-Master Multi Memory Interface (M4IF) controls memory accesses (read/write/erase/program) from one or more masters through different port interfaces to different external memory controllers ESDCTLv2, NFCv2, and WEIMv2, and to some internal memories in the system as well. i.MX51 block diagram is shown in Figure 44.1.2.

44.1.1 Overview

M4IF module is the EMIv2 module interconnect between system masters and external or internal memory controllers. M4IF module is responsible for major functions in EMIv2 module. It contains the heart of arbitration logic for the different memory interfaces. It contains the write and read buffers to support best performance data flow with minimum latency. It contains an AXI port gasket modules for each master. It manages the AXI access type and transfers it to the arbitration logic. From the arbitration it moves on to dedicated memory controller.

M4IF module is capable of handling several AXI accesses of a different type. Any EMIv2 handshake protocol between EMIv2 and system level like Low Power Mode entry and DVFS would be handled by M4IF module. M4IF would interface between the system request and the internal logic needs including memory controllers limitations.

The following sections describe the M4IF modules.

44.1.1.1 AXI Port Gasket

This is the EMIv2 interface module to AXI masters. M4IF contains 8 AXI port gaskets. This module samples the access signals, address and controls into FIFO and sorts the access to the proper destination - Fast External Memory, Slow External Memory, Internal1 Memory or Internal2 Memory channels. The module controls the AXI interface and complies to AXI bus protocol.

44.1.1.2 Dedicated Write Buffers Module

Each AXI port gasket (Master) has a Dedicated Write buffer Module. Write data from a master is sampled in the Write Buffers Module and transferred to the Memory controller upon request. The Write Buffer size is different from master to master and is calculated as follows:

buffer_size (Bytes) = buffer depth (num of slots) * 8 (max num of data per burst) * 9 Bytes (64bit-data, 8bit-strobe).

44.1.1.3 Read Shared Buffers Module

The Read Shared buffer Module serves all 8 AXI ports. M4IF contains 4 read buffers, one read buffer per channel - Slow External, Fast External, Internal1, and Internal2 Memory channels. The data from the

memory controller is written to the read buffer. The gasket provides the data back to the master when necessary.

Refer to Section 44.3.21, Buffers Size Table.

44.1.1.4 Fast Arbitration Module

The Fast Arbitration Module arbitrates master's accesses to the ESDCTL (DDR controller) by using most comprehensive information to optimize accesses to the ESDCTL, such as Page hit/miss read/write access and so on. If needed this additional information influence can be disabled and the arbitration will function in fixed priority mode, based on the user defined priority. More details on the dynamic priority arbitration can be found in the functional description section. See Section 44.3.5, Fast Arbitration Module Functional Description - 1st Degree.

44.1.1.5 Slow Arbitration Module

The Slow Arbitration Module arbitrates between the masters using a pre-defined bus division. It then sends the accesses to the WEIM memory controller and Nand-Flash controller (NFC). The division is configurable with high resolution. Further details will be provided in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division).

44.1.1.6 Internal 1 Memory Arbitration Module

The Internal 1 Arbitration Module arbitrates between the masters using a pre-defined bus division. It then sends the accesses to the internal RAM1 memory controller. The division is configurable with high resolution. Further details will be provided in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division).

44.1.1.7 Internal 2 Memory Arbitration Module

The Internal 2 Arbitration Module arbitrates between the masters using a pre-defined bus division. It then sends the accesses to the internal RAM2 memory controller. The division is configurable with high resolution. Further details will be provided in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division).

44.1.1.8 Debug Unit

The Debug Unit gives the ability to monitor several internal important EMIv2 signals to have better debug capability of the system. It is recommended that these signals would be routed out of the chip based on SoC debug definition. In addition, the debug unit would give the option to do a profiling of the arbitration unit of each channel. In that case the user would be able to trace its master bus performance and to be able to fine tune the arbitration's parameters. The debug unit contains several registers as well, so that the info can be read from the registers, observed on the pads, or analyzed by the profiling unit data. The debug unit will be described in detail in the functional section.

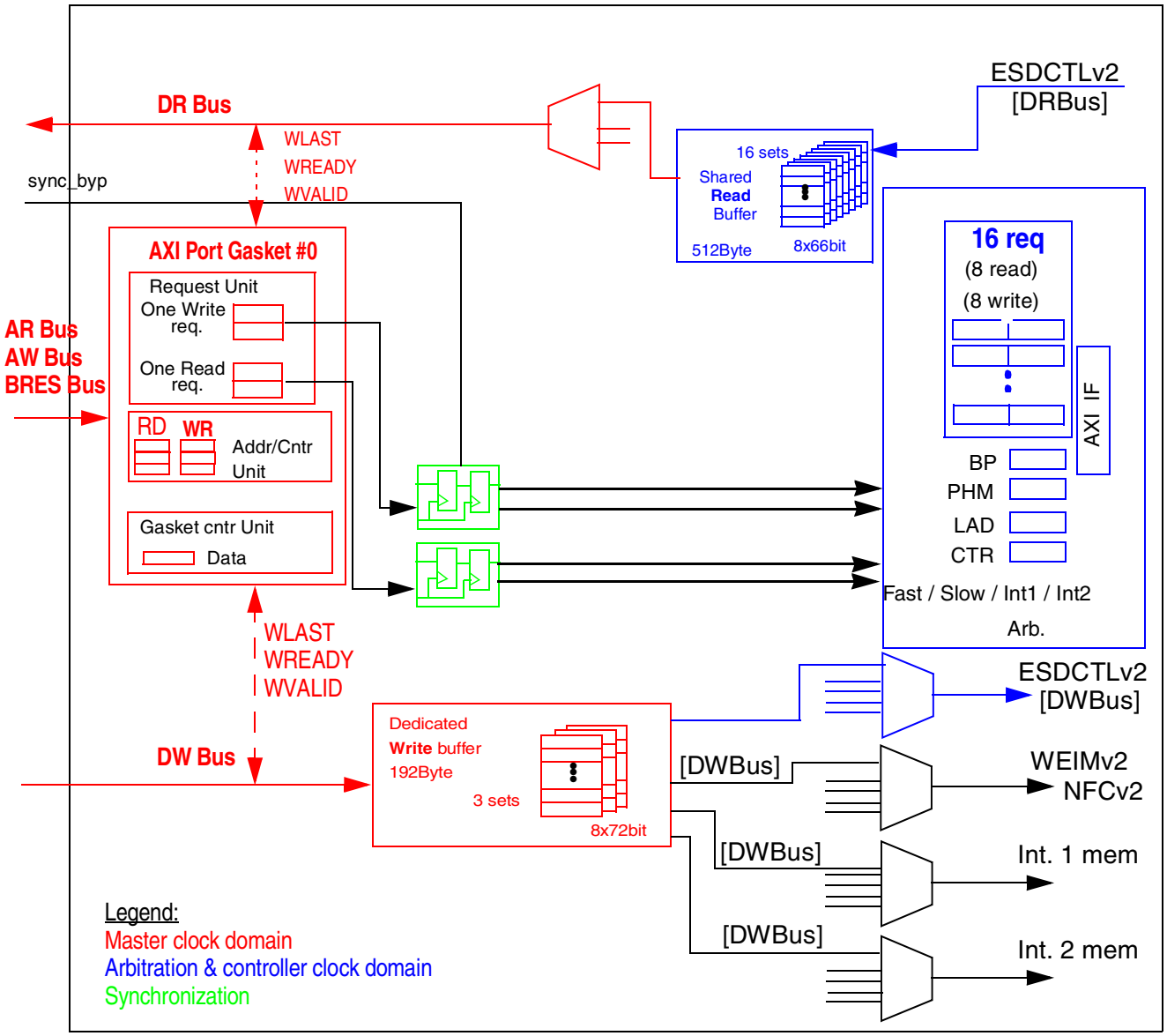


Figure 44-1. M4IF Detailed Block Diagram

44.1.2 Features

The M4IF module includes the following features:

- Supports multiple requests from up to 8 masters through different input ports interfaces. Each port can support either of the following two data width options:
  - — x32 AXI port.
  - — x64 AXI port.
- Arbitrates any master access to 5 main memory controllers, ESDCTLv2, WEIMv2, NFCv2, internal 1 salve 0 / slave 1 memory controller and internal 2 memory controller at system level.

- Supports different asynchronous master clock domain for each AXI port, using synchronization system with bypass configuration option in case clocks are aligned and balanced.
- Supports memory snooping up to two regions, for ESDCTLv2 memory region only. An example of which would be monitoring a region in external memory for write accesses:
  - The region's location is specified by a base address (from 2 KB up to 16 MB), which is divided into 64 equal segments.
  - Each segment has an access status and enable bit in the M4IF register definition.
  - M4IF generates a DMA_ACCESS for each snooping detection, total of two.
- Supports memory watermark protection for up to 8 different chip selects (2 of ESDCTL and 6 of WEIM) for preselected masters.
  - Two configurable WM configurations, for two separate masters/domains.
  - Configurable protected memory region (base address with 4 KB resolution) for each one of the watermark memory regions.
  - One status bit for each memory region that indicates security violation.
  - Maskable watermark interrupt generation capability in case of security violation for the two WM configurations.
- Separate read and write data buffer for better performance and minimum latency.
- Separate arbitration logic for fast memory interfaces - ESDCTLv2, slow memory interfaces - WEIMv2 & NFCv2, internal 1 slave 0 / slave 1 memory interfaces and for internal 2 memory interfaces like internal ROM and RAM.
- Separate clocks for each arbitration module.
- Enhanced arbitration scheme for fast channel, considering page hit/miss, last access details (read/write), fixed priority configuration,MIF4 mechanism and MIF4 prediction.
- Support AXI exclusive and lock access per arbitration path, so that locked access of one of the masters to WEIMv2 doesn't lock all EMIv2 and access to ESDCTLv2 for example can still be served.
- Bus Division arbitration scheme with programable resolution for the slow, internal memory and internal 2 memory arbitration logics.
- Support low power mode techniques so that user can put into low power entire M4IF or just part of it like slow arbitration only and so on.
- Support auto power saving features for minimum power consumption in idle mode.
- Support warm reset to allow dynamic reset configuration with minimum system impact.
- Support several debug capabilities.

44.1.3 Modes of Operation

The following sections describe the operation modes supported in M4IF module.

44.1.3.1 Normal Operating Modes

At the normal operation mode M4IF can serve each master access read or write in parallel. All clocks are enabled and READY signal of the AXI protocol are at high condition. Any access from any master can be served. For more details about normal operation mode please refer to Section 44.3, Functional Description.

44.1.3.2 Low Power Modes

Low Power mode operation is supported in M4IF. A low power mode request (lpmd) coming from the SoC would start low power mode routine. In this mode new accesses cannot pass through M4IF between a master and a memory controller, external or internal. Entering into this mode would lead the whole EMIv2 to be in Low power mode, meaning SDRAM external devices will be in self refresh mode and new accesses will not be served. All arbitration modules would finish to serve all pending requests and clean their buffers. Pending accesses on the memory controller bus will be completed.

In parallel to this routine, M4IF passes the lpmd request to the memory controllers to initialize their low power mode sequence. At the end of this process each memory controller sends back to M4IF lpmd acknowledge (lpack). M4IF, in turn, sends the Low Power Mode Acknowledge signal to the system based on its own process acknowledge and on the memory controllers‘acknowledge. This allows the system’s clock controller module to disable EMIv2 clocks without any risk.

Getting out of low power mode is done by first enabling all EMIv2 clocks and then by changing low power mode request signal to LOW state. After enabling the internal sub modules, gaskets and arbitrations, M4IF will enable new access entry to EMIv2.

In addition to the description above M4IF supports getting into low power mode by SW definition. Please refer to M4IF control register for more details.

M4IF supports partial low power mode as well. The system can enter each memory arbitration path into low power mode separately, so that one path can be in idle mode while the other is in low power mode. For example, if DDR SDRAM is currently working and no accesses to WEIM or NFC are needed, the system can set lpmd_slow signal to high so that slow path will be entered into low power mode. Other option will be to rely on auto power saving mode to allow M4IF to enter this path into low power mode automatically.

For more details on power saving mode please refer to Section 44.1.3.5, Power Saving Mode.

NOTE

In partial low power mode, it is the system’s responsibility to stop accesses to the shut-down path.

44.1.3.3 Debug Mode

44.1.3.3.1 Step By Step Mode

This mode allows the user to freeze a specific arbitration after every new transaction that is sent outside the M4IF. The user could then monitor several internal signals of M4IF (while the specific arbitration is on hold) and later write to a register that will release the current transaction and freeze the next one.

Currently, there is no option to enter this mode in the middle of a working system. Entering this mode must be done when there are no transactions in the system. Preferably immediately after reset, before any AXI accesses has been issued to M4IF/EMI.

Another way to ensure a quiet system is using the LPMD/LPACK mechanism, either by HW or SW (see Section 44.2.3.30, M4IF Control Register #0). By using the LPMD mechanism the M4IF guarantees that all pending accesses have been served.

Once the “step by step” mode has been enabled all accesses are halted. The properties of the access at the top of the PA fifo can be read via IP registers. (see Section 44.2.3.28, Step By Step Address)

In order to release the access at the top of the fifo, the appropriate bit in the MDCR should be set. Either one of the 4 bits: FSBS,SSBS,ISBS or I2SBS depending on which arbitration is being halted.

This process will repeat itself until the SBS_EN bit in the MDCR is cleared.

44.1.3.3.2 Debug Unit Functional description

For details about functionality of the Debug Unit in M4IF please refer to Section 44.3, Functional Description.

44.1.3.3.3 Debug Signals

The signal list that can be reflected outside is shown in Table 44-1:

Table 44-1. M4IF Debug Signals

Signal Name	Number of Bits	Description
master_fast	[2:0]	EMI Master number of the **selected** access at fast arbitration bus.
master_slow	[2:0]	EMI Master number of the **selected** access at slow arbitration bus.
master_int1	[2:0]	EMI Master number of the **selected** access at internal 1 memory arbitration bus.
master_int2	[2:0]	EMI Master number of the **selected** access at internal 2 memory arbitration bus.
master_id_fast	[3:0]	Master ID of the **selected** access at fast arbitration bus.
master_id_slow	[3:0]	Master ID of the **selected** access at slow arbitration bus.
master_id_int1	[3:0]	Master ID of the **selected** access at internal 1 memory arbitration bus.
master_id_int2	[3:0]	Master ID of the **selected** access at internal 2 memory arbitration bus.
id_fast	[3:0]	AXI transaction ID of the **selected** access at the fast memory arbitration bus.
id_slow	[3:0]	AXI transaction ID of the **selected** access at the slow memory arbitration bus.
id_int1	[3:0]	AXI transaction ID of the **selected** access at the internal 1 memory arbitration bus.
id_int2	[3:0]	AXI transaction ID of the **selected** access at the internal 2 memory arbitration bus.
dyn_pr_fast	[5:0]	Fast Dynamic priority of a **certain** request. (configured by MDCR[20:16])
dyn_pr_slow	[5:0]	**Equal 0**, there is no Dynamic Priority mechanism in slow arbitration
dyn_pr_int1	[5:0]	**Equal 0**, there is no Dynamic Priority mechanism in internal 1 arbitration

Table 44-1. M4IF Debug Signals (continued)

Signal Name	Number of Bits	Description
dyn_pr_int2	[5:0]	**Equal 0**, there is no Dynamic Priority mechanism in internal 2 arbitration
acc_type_fast	—	access type of the **selected** access at fast arbitration bus.
acc_type_slow	—	access type of the **selected** access at slow arbitration bus.
acc_type_int1	—	access type of the **selected** access at internal 1 memory arbitration bus.
acc_type_int2	—	access type of the **selected** access at internal 2 memory arbitration bus.
addr_fast	[31:0]	AXI ADDRESS of the **selected** access at fast arbitration bus.
addr_slow	[31:0]	AXI ADDRESS of the **selected** access at slow arbitration bus.
addr_int1	[31:0]	AXI ADDRESS of the **selected** access at internal 1 memory arbitration bus.
addr_int2	[31:0]	AXI ADDRESS of the **selected** access at internal 2 memory arbitration bus.

The selected arbitration's signals are routed outside via a bus called: **ipp_do_emi_debug**. Only one arbitration can be viewed on the debug bus.

They are ordered on this bus in the following way:

ipp_do_emi_debug[50:0] = {valid_strobe,master,master_id,id,dyn_pr,access_type,addr}

where,

- valid_strobe — a signal to indicate that a valid request is selected, this signal will be 3 clocks wide. The clock depends on the arbitration which is selected.
- master — EMIv2 master port (0:7), 3 bits wide.(Selected between: master_fast, master_slow, master_int1, master_int2)
- master_id — system MASTER ID, 4 bits wide.
- id — AXI transaction ID, 4 bits wide.
- dyn_pr — the updated dynamic priority of a request, 6 bits wide.
- access_type — write (0) or read (1), 1 bit.
- addr — the address that is accessed (base address), 32 bits wide.

44.1.3.4 DVFS

DVFS is supported in M4IF. When DVFS request is high, M4IF sends DVFS request to the memory controllers while it keeps working normally. Reacting to DVFS request, the memory controllers would change their READY signals state to low and start DVFS routine while M4IF will keep working as in normal mode. Since READY signals of the memory controllers is low, no new access will pass from M4IF to the memory controller but only piped access in the memory controller will be cleaned. At the end, the memory controller pipeline will be cleaned and a DVFS ack signals will be sent back to M4IF. M4IF will then send a DVFS ack signal to the system to allow changing frequency.

In addition to the description above M4IF supports DVFS request to be enabled by register configuration. Please refer to DVFS register for more details.

44.1.3.5 Power Saving Mode

44.1.3.5.1 Arbitration Power Saving

In order to support power saving mode to reduce power consumption of the system, M4IF would be able to detect whether there is a request pending on any of the arbitration modules or not. The power saving register defines the time period that should pass from the last served request in a certain arbitration. If during the specified time period no request was observed on the arbitration the arbitration module would send low power mode request to its memory controller and would enter itself to low power mode as well. The power saving mode is defined and functioned separately for each arbitration. In that case fast arbitration can be in normal mode while the slow arbitration can be in powder saving mode. Getting out of this mode would be started automatically when a new request to the specific arbitration is issued by one of the masters. It is important to mention that it would take few cycles to serve the request since getting out of power saving mode routine is takes few cycles at M4IF as well as in the memory controller.

44.1.3.5.2 Gasket Power Saving

Power saving feature is also supported for each gasket individually. The gasket will detect a number of cycles that it were idle. Upon reaching this configurable number of cycles, the gasket will de-assert its clock gating enable, and almost all the FFs inside will stop. There will be a few FFs still running on a free running clock, in order to be able to wake from this mode. In this mode the 3 ready signals: AWREADY, ARREADY and WREADY will be de-asserted until an access is issued to this gasket. When detecting a new access (AWVALID, ARVALID or WVALID asserted), the gasket will assert its clock enable signal (in the next cycle). There is a minimum number of cycles that the READY signals has to be de-asserted after the clock enable is re-asserted. Please note these cycles are of the **master clock**. For each master the clock period is different. The number of cycles is configurable via a register. The number of idle cycles that will initiate this process is also a configurable number. Refer to Section 44.2.3.1, Power Saving Masters 0 for further details.

Figure 44-2 illustrates the behaviour of the clock enable and of the ready signals upon exiting from power saving. In this diagram the “ready off cycles” (ROC) is at its default value 3.

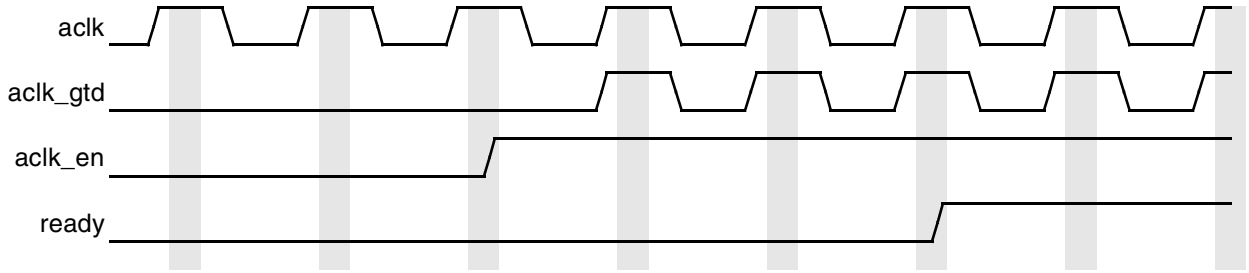


Figure 44-2. Behaviour of Clock Enable and Ready Signals

44.2 Memory Map and Register Definition

The sections below describe the register definition of M4IF module. It contains all information on registers definition like watermark and control registers. It contains all user programable master priority register per arbitration path as well as priority specific definition for fast arbitration.

44.2.1 Memory Map

M4IF memory area is in the IP memory region dedicated for its registers. M4IF registers are located at 0xBASE+0x0000 - 0xBASE+0x0FFF.

44.2.2 Register Summary

The conventions in Figure 44-3 and Table 44-2 serve as a key for the register summary and individual register diagrams.

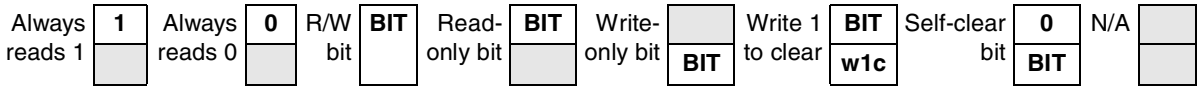


Figure 44-3. Key to M4IF Register Fields

Table 44-2 provides a key for register figures and tables and the register summary.

Table 44-2. M4IF Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
w1s	Write one to set. A status bit that can be read, and is set by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 44-3 shows the M4IF register summary.

Table 44-3. M4IF Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (PSM0)	R	M1_PST									M1_WIS	M1_RIS	M1_PSS	M1_ROC			M1_PSD
	W																
	R	M0_PST									M0_WIS	M0_RIS	M0_PSS	M0_ROC			M0_PSD
	W																
0xBASE+0x004 (PSM1)	R	M3_PST									M3_WIS	M3_RIS	M3_PSS	M3_ROC			M3_PSD
	W																
	R	M2_PST									M2_WIS	M2_RIS	M2_PSS	M2_ROC			M2_PSD
	W																
0xBASE+0x00 0C Reserved	R																
	W																
	R																
	W																
0xBASE+0x01 0 Reserved	R																
	W																
	R																
	W																
0xBASE+0x01 4 Reserved	R																
	W																
	R																
	W																
0xBASE+0x01 8 (MDSR6)	R												REQ_ACC				
	W																
	R	REQ_ACC															
	W																
0xBASE+0x01 C (MDSR7)	R	STR															
	W																
	R	STR															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x020 (MDSR8)	R	STM															
	W																
	R	STM															
	W																
0xBASE+0x024 (MDSR0)	R							MAX_TIME									
	W																
	R											MAX_DPR					
	W																
0xBASE+0x028 (MDSR1)	R									TOT_ACC							
	W																
	R	TOT_ACC															
	W																
0xBASE+0x02C (MDSR2)	R	M1_ACC															
	W																
	R	M0_ACC															
	W																
0xBASE+0x030 (MDSR3)	R	M3_ACC															
	W																
	R	M2_ACC															
	W																
0xBASE+0x034 (MDSR4)	R	M5_ACC															
	W																
	R	M4_ACC															
	W																
0xBASE+0x038 (MDSR5)	R	M7_ACC															
	W																
	R	M6_ACC															
	W																
0xBASE+0x003C (Reserved)	R																
	W																
	R																
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x040 (FBPM0)	R						FBPM_M3								FBPM_M2		
	W																
	R						FBPM_M1								FBPM_M0		
	W																
0xBASE+0x044 (FBPM1)	R						FBPM_M7								FBPM_M6		
	W																
	R						FBPM_M5								FBPM_M4		
	W																
0xBASE+0x048 (MIF4)	R											MIF4_PAG_HIT			MIF4_ACC_HIT		
	W																
	R					MIF4_DYN_JMP				MIF4_DYN_MAX					MIF4_GUARD		
	W																
0xBASE+0x04C (SBAR0)	R	SWBA															
	W																
	R	SWBA0								SSW0			SWSZ0				SE0
	W																
0xBASE+0x050 (SERL0)	R	SSEL0															
	W																
	R	SSEL0															
	W																
0xBASE+0x054 (SERH0)	R	SSEH0															
	W																
	R	SSEH0															
	W																
0xBASE+0x058 (SSRL0)	R	SSSL0															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	SSSL0															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x05C (SSRH0)	R	SSSH0															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	SSSH0															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x060 (SBAR1)	R	SWBA1															
	W																
	R	SWBA1								SSW1			SWSZ1				SE1
	W																
0xBASE+0x064 (SERL1)	R	SSEL1															
	W																
	R	SSEL1															
	W																
0xBASE+0x068 (SERH1)	R	SSEH1															
	W																
	R	SSEH1															
	W																
0xBASE+0x06C (SSRL1)	R	SSSL1															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	SSSL1															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x070 (SSRH1)	R	SSSH1															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
	R	SSSH1															
	W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x074 (I2ULA)	R												I2L5			I2L4M47	
	W																
	R	I2L4M47	I2L4M03			I2L3M67			I2L3M45			I2L3M23			I2L3M01		
	W																
0xBASE+0x078 (I2ACR)	R																
	W																
	R								I2RDT								
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x07C (RINT2)	R																
	W																
	R			I2DVACK	I2LPACK	I2PSS	I2PST								I2DVFS	I2LPMD	DI2PS
	W																
0xBASE+0x0080 (Reserved)	R																
	W																
	R																
	W																
0xBASE+0x084 (SBS0)	R	SBS_ADDR															
	W																
	R	SBS_ADDR															
	W																
0xBASE+0x088 (SBS1)	R			SBS_VLD	SBS_TYPE	SBS_LEN			SBS_SIZE		SBS_BURST		PROT_SBS			LOCK_SBS	
	W																
	R	SBS_CACHE				SBS_AXI_ID				SBS_MASTER_ID				SBS_MASTER			SBS_END
	W																
0xBASE+0x08C (MCR0)	R	SW_RST				FDVACK	SDVACK	IDVACK	DVACK	FLPACK	SLPACK	ILPACK	LPACK		FPSS	SPSS	IPSS
	W																
	R	EAS3	EAS2	EAS1	EAS0	FDVFS	SDVFS	IDVFS	DVFS	FLMPD	SLPMD	ILPMD	LPMD		DFPS	DSPS	DIPS
	W													*			
0xBASE+0x090 (MCR1)	R	EAS7	EAS6	EAS5	EAS4			IPST									SPST
	W																
	R	SPST								FPST							
	W																
0xBASE+0x094 (MDCR)	R	DBG_EN				DBG_RST							DDPT	DDPM			
	W																
	R	SBS_EN	FSBS	SSBS	ISBS	I2SBS		RARB								VARB	
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xBASE+0x098 (FACR)	R																
	W																
	R								FRDT	FDPUR							FDPE
	W																
0xBASE+0x09C (FPWC)	R										FAHR			FPHR			
	W																
	R						FPCR				FPAR			FDPM			
	W																
0xBASE+0x0A0 (SACR)	R																
	W																
	R								SRDT								
	W																
0xBASE+0x0A4 (PSM2)	R	M5_PST									M5_WIS	M5_RIS	M5_PSS	M5_ROC			M5_PSD
	W																
	R	M4_PST									M4_WIS	M4_RIS	M4_PSS	M4_ROC			M4_PSD
	W																
0xBASE+0x0A8 (IACR)	R																
	W																
	R								IRDT								
	W																
0xBASE+0x0AC (PSM3)	R	M7_PST									M7_WIS	M7_RIS	M7_PSS	M7_ROC			M7_PSD
	W																
	R	M6_PST									M6_WIS	M6_RIS	M6_PSS	M6_ROC			M6_PSD
	W																
0xBASE+0x0B0 (FULA)	R												FL5			FL4M47	
	W																
	R	FL4M47	FL4M03			FL3M67			FL3M45			FL3M23			FL3M01		
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0B4 (SULA)	R												SL5			SL4M47	
	W																
	R	SL4M47	SL4M03			SL3M67			SL3M45			SL3M23			SL3M01		
	W																
0xBASE+0x0B8 (IULA)	R												IL5			IL4M47	
	W																
	R	IL4M47	IL4M03			IL3M67			IL3M45			IL3M23			IL3M01		
	W																
0xBASE+0x0BC (FDPS)	R			FDP4						FDP3						FDP2	
	W																
	R	FDP2				FDP1						FDP0					
	W																
0xBASE+0x0C0 (FDPC)	R								FD4_MAS			FD4_RW		FD3_MAS			FD3_RW
	W																
	R		FD2_MAS			FD2_RW			FD1_MAS		FD1_RW		FD0_MAS			FD0_RW	
	W																
0xBASE+0x0C4 (MLEN)	R																
	W																
	R	fb816	sb816	i1b816	i2b816					m7len	m6len	m5len	m4len	m3len	m2len	m1len	m0len
	W	w1c	w1c	w1c	w1c					w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x0C8 (Reserved)	R																
	W																
	R																
	W																
0xBASE+0x0CC (Reserved)	R																
	W																
	R																
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xBASE+0x0D0 (Reserved)	R																
	W																
	R																
	W																
0xBASE+0x0D4 (WMSA0_0)	R	WE0_0															
	W																
	R	WMS0_0															
	W																
0xBASE+0x0D8 (WMSA0_1)	R	WE0_1															
	W																
	R	WMS0_1															
	W																
0xBASE+0x0DC (WMSA0_2)	R	WE0_2															
	W																
	R	WMS0_2															
	W																
0xBASE+0x0E0 (WMSA0_3)	R	WE0_3															
	W																
	R	WMS0_3															
	W																
0xBASE+0x0E4 (WMSA0_4)	R	WE0_4															
	W																
	R	WMS0_4															
	W																
0xBASE+0x0E8 (WMSA0_5)	R	WE0_5															
	W																
	R	WMS0_5															
	W																
0xBASE+0x0EC (WMSA0_6)	R	WE0_6															
	W																
	R	WMS0_6															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0xBASE+0x0F0 (WMSA0_7)	R	WE0_7															
	W																
	R	WMS0_7															
	W																
0xBASE+0x0F4 (WMEA0_0)	R																
	W																
	R	WME0_0															
	W																
0xBASE+0x0F8 (WMEA0_1)	R																
	W																
	R	WME0_1															
	W																
0xBASE+0x0FC (WMEA0_2)	R																
	W																
	R	WME0_2															
	W																
0xBASE+0x100 (WMEA0_3)	R																
	W																
	R	WME0_3															
	W																
0xBASE+0x104 (WMEA0_4)	R																
	W																
	R	WME0_4															
	W																
0xBASE+0x108 (WMEA0_5)	R																
	W																
	R	WME0_5															
	W																
0xBASE+0x10C (WMEA0_6)	R																
	W																
	R	WME0_6															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x110 (WMEA0_7)	R																
	W																
	R	WME0_7															
	W																
0xBASE+0x114 (WMIS0)	R	WIE0		FVMID_0											SVMID_0		
	W																
	R	SVMID_0								WS0_7	WS0_6	WS0_5	WS0_4	WS0_3	WS0_2	WS0_1	WS0_0
	W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x118 (FWMVA0)	R	FWAVA0															
	W																
	R	FWAVA0															
	W																
0xBASE+0x11C (SWMVA0)	R	SWAVA0															
	W																
	R	SWAVA0															
	W																
0xBASE+0x120 (WMSA1_0)	R	WE1_0															
	W																
	R	WMS1_0															
	W																
0xBASE+0x124 (WMSA1_1)	R	WE1_1															
	W																
	R	WMS1_1															
	W																
0xBASE+0x128 (WMSA1_2)	R	WE1_2															
	W																
	R	WMS1_2															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x12C (WMSA1_3)	R	WE1_3															
	W																
	R	WMS1_3															
	W																
0xBASE+0x130 (WMSA1_4)	R	WE1_4															
	W																
	R	WMS1_4															
	W																
0xBASE+0x134 (WMSA1_5)	R	WE1_5															
	W																
	R	WMS1_5															
	W																
0xBASE+0x138 (WMSA1_6)	R	WE1_6															
	W																
	R	WMS1_6															
	W																
0xBASE+0x13C (WMSA1_7)	R	WE1_7															
	W																
	R	WMS1_7															
	W																
0xBASE+0x140 (WMEA1_0)	R																
	W																
	R	WME1_0															
	W																
0xBASE+0x144 (WMEA1_1)	R																
	W																
	R	WME1_1															
	W																
0xBASE+0x148 (WMEA1_2)	R																
	W																
	R	WME1_2															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x14C (WMEA1_3)	R																
	W																
	R	WME1_3															
	W																
0xBASE+0x150 (WMEA1_4)	R																
	W																
	R	WME1_4															
	W																
0xBASE+0x154 (WMEA1_5)	R																
	W																
	R	WME1_5															
	W																
0xBASE+0x158 (WMEA1_6)	R																
	W																
	R	WME1_6															
	W																
0xBASE+0x15C (WMEA1_7)	R																
	W																
	R	WME1_7															
	W																
0xBASE+0x160 (WMIS1)	R	WIE1		FVMID_1											SVMID_1		
	W																
	R	SVMID_1								WS1_7	WS1_6	WS1_5	WS1_4	WS1_3	WS1_2	WS1_1	WS1_0
	W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x164 (FWMVA1)	R	FWAVA1															
	W																
	R	FWAVA1															
	W																

Table 44-3. M4IF Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x168 (SWMVA1)	R	SWAVA1															
	W																
	R	SWAVA1															
	W																

44.2.3 Register Descriptions

This section consists of M4IF register descriptions in address order. Please refer to next sections for detailed description of each register.

44.2.3.1 Power Saving Masters 0

The PSM0 determines the power saving features for masters #0 and #1.

Address 0xBASE+0x000 (PSM0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M1_PST									M1_WIS	M1_RIS	M1_PSS	M1_ROC			M1_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M0_PST									M0_WIS	M0_RIS	M0_PSS	M0_ROC			M0_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

Figure 44-4. Power Saving Masters 0 Register

Detailed description of Power Saving Masters 0 register is shown in Table 44-4.

Table 44-4. Power Saving Masters 0 Register Field Descriptions

Field	Description
31-24 M1_PST	master #1 Power Saving Timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
23	Reserved.
22 M1_WIS	Master #1 Write Idle Status.This read only bit indicates whether master #1 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
21 M1_RIS	Master #1 Read Idle Status.This read only bit indicates whether master #1 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
20 M1_PSS	Master #1 Power Saving Status. This read only bit indicates whether master #1 gasket is in power saving mode. 0 - not in power saving 1 - power saving
19-17 M1_ROC	master #1 Ready Off Cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. 3'b001 - 3'b111 (3'b000 is a forbidden value)
16 M1_PSD	master #1 Power Saving Disable. 0 - power saving enabled 1 - power saving disabled (default)
15-8 M0_PST	master #0 power saving timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
7	Reserved.
6 M0_WIS	Master #0 Write Idle Status.This read only bit indicates whether master #0 write request buffer is idle (empty) or not. 0 - idle 1 - not idle

Table 44-4. Power Saving Masters 0 Register Field Descriptions (continued)

Field	Description
5 M0_RIS	Master #0 Read Idle Status.This read only bit indicates whether master #0 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
4 M0_PSS	Master #0 Power Saving Status. This read only bit indicates whether master #0 gasket is in power saving mode. 0 - not in power saving 1 - power saving
3-1 M0_ROC	master #0 Ready Off Cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. Default value is 3 cycles. 3'b001 - 3'b111 (3'b000 is a forbidden value)
0 M0_PSD	master #0 Power Saving Disable. 0 - power saving enabled 1 - power saving disabled (default)

44.2.3.2 Power Saving Masters 1

The PSM1 determines the power saving features for masters #2 and #3.

Address 0xBASE+0x004 (PSM1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M3_PST									M3_WIS	M3_RIS	M3_PSS	M3_ROC			M3_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M2_PST									M2_WIS	M2_RIS	M2_PSS	M2_ROC			M2_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

Figure 44-5. Power Saving Masters 1 Register

Detailed description of Power Saving Masters 1 register is shown in Table 44-4.

Table 44-5. Power Saving Masters 1 Register Field Descriptions

Field	Description
31-24 M3_PST	master #3 Power Saving Timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
23	Reserved.
22 M3_WIS	Master #3 Write Idle Status.This read only bit indicates whether master #3 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
21 M3_RIS	Master #3 Read Idle Status.This read only bit indicates whether master #3 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
20 M3_PSS	Master #3 Power Saving Status. This read only bit indicates whether master #3 gasket is in power saving mode. 0 - not in power saving 1 - power saving
19-17 M3_ROC	master #3 Clock Off Cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. 3'b001 - 3'b111 (3'b000 is a forbidden value)
16 M3_PSD	master #3 Power Saving Disable. 0 - power saving enabled 1 - power saving disabled (default)
15-8 M2_PST	master #2 Power Saving Timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
7	Reserved.
6 M2_WIS	Master #2 Write Idle Status.This read only bit indicates whether master #2 write request buffer is idle (empty) or not. 0 - idle 1 - not idle

Table 44-5. Power Saving Masters 1 Register Field Descriptions (continued)

Field	Description
5 M2_RIS	Master #2 Read Idle Status.This read only bit indicates whether master #2 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
4 M2_PSS	Master #2 Power Saving Status. This read only bit indicates whether master #2 gasket is in power saving mode. 0 - not in power saving 1 - power saving
3-1 M2_ROC	master #2 Ready Off Cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. Default value is 3 cycles. 3'b001 - 3'b111 (3'b000 is a forbidden value)
0 M2_PSD	master #2 Power Saving Disable. 0 - power saving enabled 1 - power saving disabled (default)

44.2.3.3 M4IF Debug Status Register #6

The MDSR6 reflects the total number of accesses made by a specific requesting the desired arbitration. The arbitration can be fast, slow,intr1 or intr2. Please refer to Section 44.2.3.32, M4IF Debug Control Register.

Address 0xBASE+0x018 (MDSR6) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												REQ_ACC				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-6. M4IF Debug Status Register #6 Register

Detailed description of M4IF Debug Status Register #6 register is shown in Table 44-6.

Table 44-6. M4IF Debug Status Register #6 Register Field Descriptions

Field	Description
31-21	Reserved.
20-0 REQ_ACC	Number of accesses made by a specific request to a specfic arbitarion. The request and arbitration are configured in the MDCR register.

44.2.3.4 M4IF Debug Status Register #7

The MDSR7 reflects the the sum of time all the requests of a specific kind were pending in the desired arbitration. The arbitration can be fast, slow,intr1 or intr2 and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x01C (MDSR7) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 44-7.

Figure 44-7. M4IF Debug Status Register #6 Register

Detailed description of M4IF Debug Status Register #7 register is shown in Table 44-8.

Table 44-8. M4IF Debug Status Register #7 Register Field Descriptions

Field	Description
31-0 STR	Sum Time Request. These bits reflect the sum of time (in arbitration clock cycles) all the requests of a specific kind were pending at a specfic arbitation. (The arbitration is configured in the MDCR register)

44.2.3.5 M4IF Debug Status Register #8

The MDSR8 reflects the the sum of time all the requests of a specific master were pending in the desired arbitration. The arbitration can be fast, slow,intr1 or intr2 and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x020 (MDSR8) Access: User read only

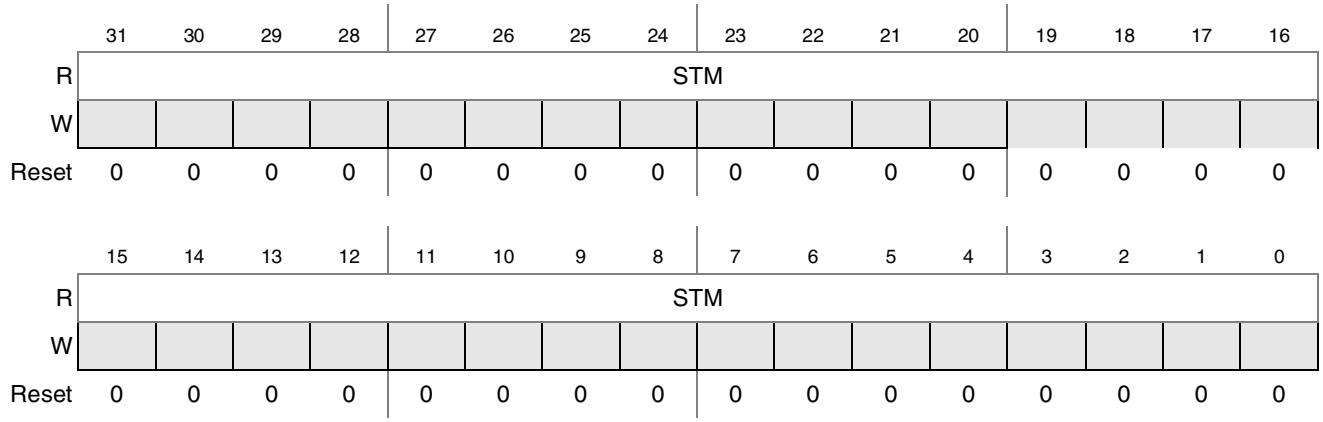


Figure 44-8. M4IF Debug Status Register #8 Register

Detailed description of M4IF Debug Status Register #8 register is shown in Table 44-9.

Table 44-9. M4IF Debug Status Register #8 Register Field Descriptions

Field	Description
31-0 STM	Sum Time Master. These bits reflect the sum of time (in arbitration clock cycles) all the requests of a specific master were pending at a specific arbitration. (The arbitration is configured in the MDCR register)

44.2.3.6 M4IF Debug Status Register #0

The MDSR0 reflects the maximum values of the chosen dynamic priority and the “time for bus” of a master. The desired dynamic priority is configured in the MDCR. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x024 (MDSR0) Access: User read only

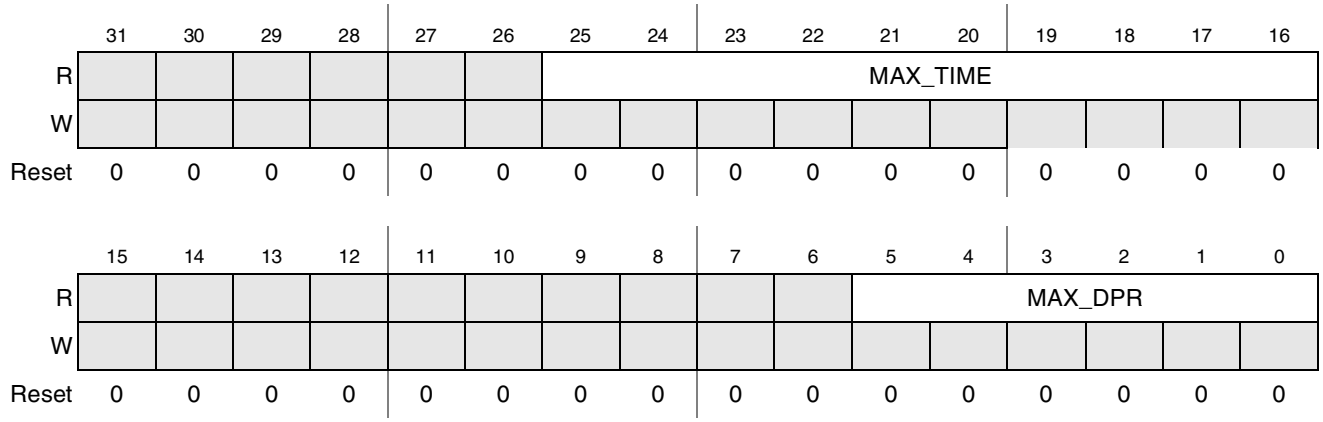


Figure 44-9. M4IF Debug Status Register #0 Register

Detailed description of M4IF Debug Status Register #0 register is shown in Table 44-10.

Table 44-10. M4IF Debug Status Register #0 Register Field Descriptions

Field	Description
31-26	Reserved.
25-16 MAX_TIME	Maximum Time for Bus. This field represents the maximum number of cycles the selected master (see DDPM field in MDCR) was pending on the bus.
15-6	Reserved.
5-0 MAX_DPR	Maximum Dynamic Priority. These bits reflect the maximum value of the dynamic priority which was selected by the MDCR.

44.2.3.7 M4IF Debug Status Register #1

The MDSR1 reflects the total number of accesses that were made through the desired arbitration. The arbitration can be fast, slow or intr, and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x028 (MDSR1) Access: User read only

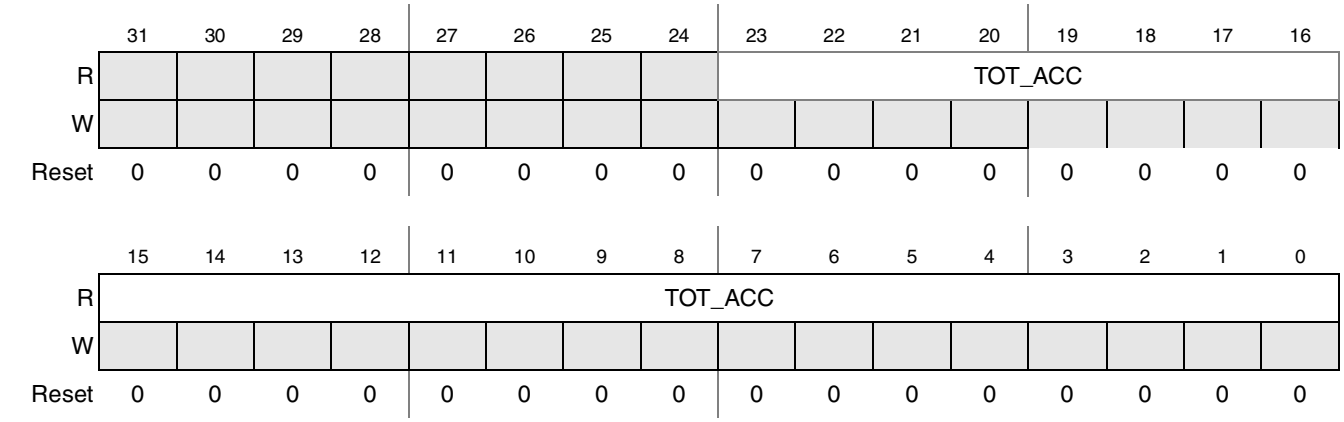


Figure 44-10. M4IF Debug Status Register #1 Register

Detailed description of M4IF Debug Status Register #1 register is shown in Table 44-11

Table 44-11. M4IF Debug Status Register #1 Register Field Descriptions

Field	Description
31-24	Reserved.
23-0 TOT_ACC	Total Accesses. These bits reflect the total number of accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.

44.2.3.8 M4IF Debug Status Register #2

The MDSR2 reflects the number of groups of 32 accesses (resolution is of 32 accesses) that master0 and master1 made through the desired arbitration. The arbitration can be fast, slow,int1 or int2 and i configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x02C (MDSR2) Access: User read only

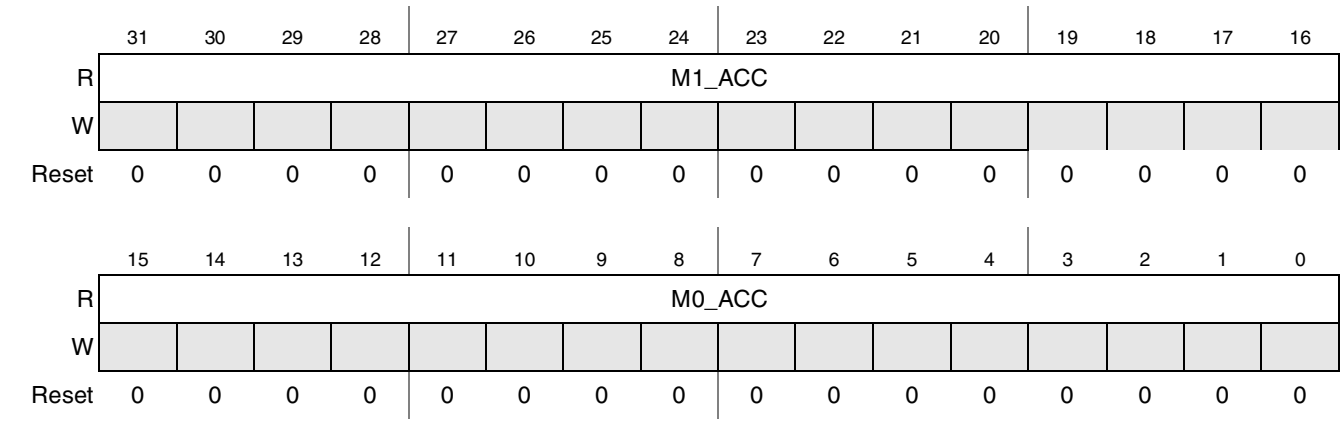


Figure 44-11. M4IF Debug Status Register #2 Register

Detailed description of M4IF Debug Status Register #2 register is shown in Table 44-12

Table 44-12. M4IF Debug Status Register #2 Register Field Descriptions

Field	Description
31-16 M1_ACC	Master1 Accesses. These bits reflect the total number of master1 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.
15-0 M0_ACC	Master0 Accesses. These bits reflect the total number of master0 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.

44.2.3.9 M4IF Debug Status Register #3

The MDSR3 register reflects the number of groups of 32 accesses (resolution is of 32 accesses) that master2 and master3 made through the desired arbitration. The arbitration can be fast, slow or internal and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x030 (MDSR3) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M3_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M2_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-12. M4IF Debug Status Register #3 register.

Detailed description of M4IF Debug Status Register #3 register is shown in Table 44-13

Table 44-13. M4IF Debug Status Register #3 Register Field Description

Field	Description
31-16 M3_ACC	Master3 Accesses. These bits reflect the total number of master3 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.
15-0 M2_ACC	Master2 Accesses. These bits reflect the total number of master2 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.

44.2.3.10 M4IF Debug Status Register #4

The MDSR4 reflects the number of groups of 32 accesses (resolution is of 32 accesses) that master4 and master5 made through the desired arbitration. The arbitration can be fast, slow or intr and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x034 (MDSR4) Access: User read only

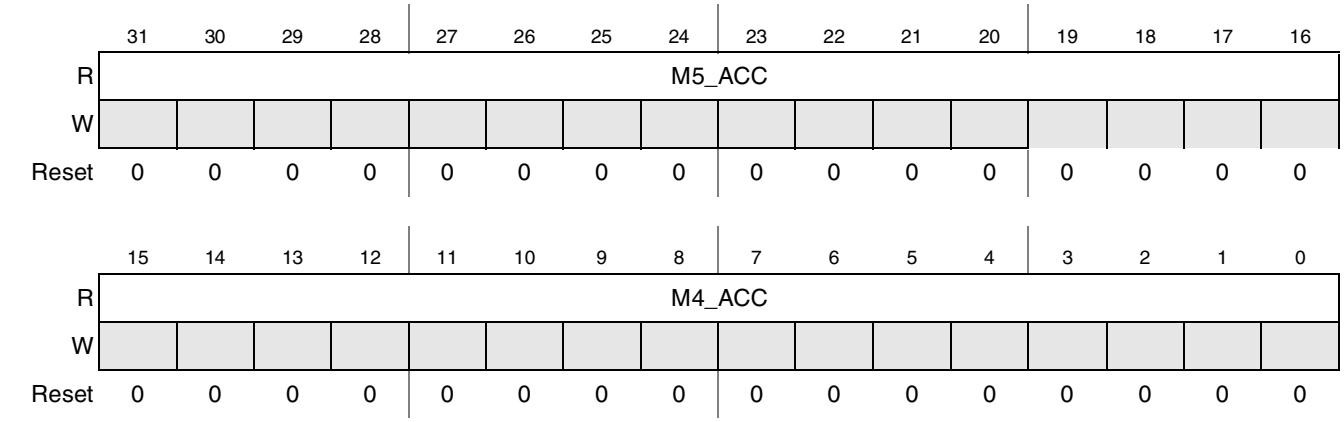


Figure 44-13. M4IF Debug Status Register #4 register.

Detailed description of M4IF Debug Status Register #3 register is shown in Table 44-14

Table 44-14. M4IF Debug Status Register #4 Register Field Description

Field	Description
31-16 M5_ACC	Master5 Accesses. These bits reflect the total number of master5 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.
15-0 M4_ACC	Master4 Accesses. These bits reflect the total number of master4 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.

44.2.3.11 M4IF Debug Status Register #5

The MDSR5 reflects the number of groups of 32 accesses (resolution is of 32 accesses) that master6 and master7 made through the desired arbitration. The arbitration can be fast, slow or intr and is configured in the MDCR register. Please refer to Section 44.2.3.32, M4IF Debug Control Register

Address 0xBASE+0x038 (MDSR5) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M7_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M6_ACC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-14. M4IF Debug Status Register #5 register.

Detailed description of M4IF Debug Status Register #3 register is shown in Table 44-15

Table 44-15. M4IF Debug Status Register #5 Register Field Description

Field	Description
31-16 M7_ACC	Master7 Accesses. These bits reflect the total number of master7 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.
15-0 M6_ACC	Master6 Accesses. These bits reflect the total number of master6 accesses that were made to a specific arbitration. The selected arbitration is configured in the MDCR.

44.2.3.12 F_Basic Priority Reg #0

FBPM0 register contains the basic priority value for masters 0,1,2,3 for the fast arbitration logic only. Different register are used for slow arbitration and internal memory arbitration logic.

Address 0xBASE+0x040 (FBPM0) Access: User read-write

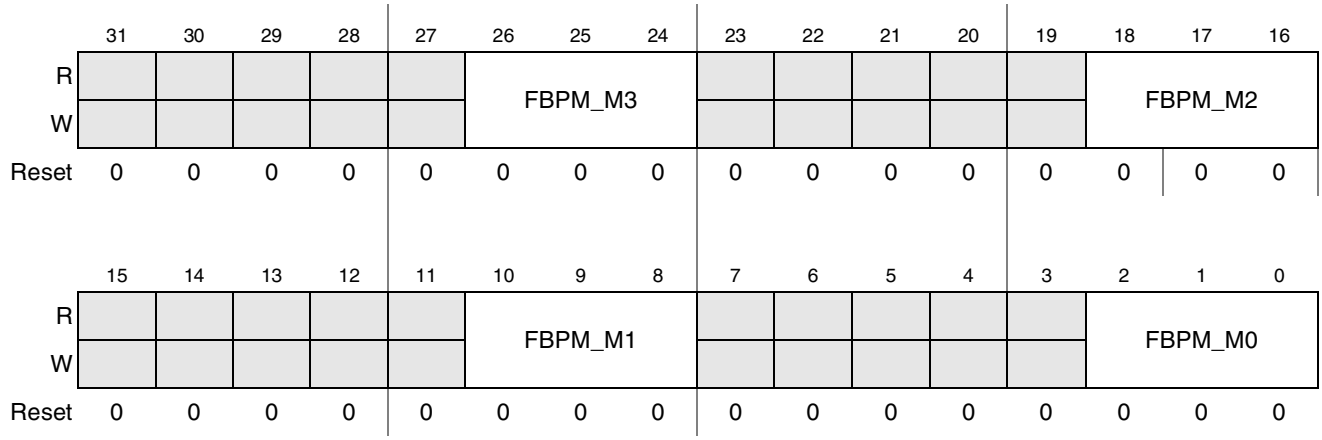


Figure 44-15. F_Basic Priority Reg #0 Register

Detailed description of F_Basic Priority Reg #0 register is shown in Table 44-16.

Table 44-16. F_Basic Priority Reg #0 Register Field Descriptions

Field	Description
31-27	Reserved.
26-24 FBPM_M3	User defined value. These bits represent the basic priority value as configured by the user to master 3 000 encoding 0 001 encoding 1 ... ... 111 encoding 7
23-19	Reserved
18-16 FBPM_M2	User defined value. These bits represent the basic priority value as configured by the user to master 2. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7
15-11	Reserved
10-8 FBPM_M1	User defined value. These bits represent the basic priority value as configured by the user to master 1. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7

Table 44-16. F_Basic Priority Reg #0 Register Field Descriptions (continued)

Field	Description
7-3	Reserved
2-0 FBPM_M0	User defined value. These bits represent the basic priority value as configured by the user to master 0. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7

44.2.3.13 F_Basic Priority Reg #1

FBPM1 register contains the basic priority value for masters 4,5,6,7 for the fast arbitration logic only. Different register are used for slow arbitration and internal memory arbitration logic.

Address 0xBASE+0x044 (FBPM1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R						FBPM_M7								FBPM_M6		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R						FBPM_M4								FBPM_M4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-16. F_Basic Priority Reg #1 Register

Detailed description of F_Basic Priority Reg #1 register is shown in Table 44-17.

Table 44-17. F_Basic Priority Reg #1 Register Field Descriptions

Field	Description
31-27	Reserved.
26-24 FBPM_M7	User defined value. These bits represent the basic priority value as configured by the user to master 7 000 encoding 0 001 encoding 1 ... ... 111 encoding 7
23-19	Reserved

Table 44-17. F_Basic Priority Reg #1 Register Field Descriptions (continued)

Field	Description
18-16 FBPM_M6	User defined value. These bits represent the basic priority value as configured by the user to master 6. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7
15-11	Reserved
10-8 FBPM_M5	User defined value. These bits represent the basic priority value as configured by the user to master 5. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7
7-3	Reserved
2-0 FBPM_M4	User defined value. These bits represent the basic priority value as configured by the user to master 4. 000 encoding 0 001 encoding 1 ... ... 111 encoding 7

44.2.3.14 MIF4 Scoring Register

This register determines the values of the weights used for the MIF4 arbitration engine.

Address 0xBASE+0x048 (MIF4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R											MIF4_PAG_HIT			MIF4_ACC_HIT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					MIF4_DYN_JMP				MIF4_DYN_MAX				MIF4_GUARD			
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	1	0	1

Figure 44-17. MIF4 Scoring Register Register

Detailed description of MIF4 Scoring Register register is shown in Table 44-18.

Table 44-18. MIF4 Scoring Register Register Field Descriptions

Field	Description
31-20	Reserved
21-19 MIF4_PAG_HIT	MIF4 Page Hit Rate. This value will be added by the mif4 arbitration logic to any pending request that is considered as page hit. Default value of MIF4_PAG_HIT is 0x00100 - encoding 4.
18-16 MIF4_ACC_HIT	MIF4 Access Hit Rate. Any pending access that is considered as page hit is checked to be a consecutive access type or not. If it is a consecutive access than the weighting value of this request will increase by MIF4_ACC_HIT value. Default value of is MIF4_ACC_HIT 0x0010 - encoding 2.
11-8 MIF4_DYN_JMP	MIF4 Dynamic Jump. Each time a request is being popped outside the MIF4 arbitration, the other pending requests will increase there dynamic score value by MIF4_DYN_JMP value. Default MIF4_DYN_JMP value is 0x0001 - encoding 1
7-4 MIF4_DYN_MAX	MIF4 Dynamic Maximum. MIF4_DYN_MAX is the maximum dynamic score value that each request inside the MIF4 mechanism can get. Default MIF4_DYN_MAX value is 0x1000 - encoding 8
3-0 MIF4_GUARD	MIF4 Guard. After a request reached the maximum dynamic score value, it will wait another MIF4_GUARD cycles and then force the request to pop outside the MIF4 arbitration. If MIF4_GUARD equals 0, this guarding mechanism is disabled. Default MIF4_GUARD value is 0x0101 - encoding 5

44.2.3.15 Snooping Base Address Register #0

The SBAR0 register contains the snooping window base address for region 0, the size of snooping window and the snooping control bit fields which are used by the fast arbitration logic to monitor the write access to CSD0 and CSD1.

Address 0xBASE+0x04C (SBAR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SWBA0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SWBA0								SSW0				SWSZ0			SE0
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figure 44-18. Snooping Base Address Register #0

Details on the register fields are shown below in Table 44-19

Table 44-19. Snooping Base Address Register #0 Field Descriptions

Field	Description
31–11 SWBA0	Snooping Window Base Address. This field defines the snooping window base address (upper 21 bits of address) to be monitored by the M3IF. M3IF monitors write accesses to the memory region above the base address window. See description in Table 44-20
10–8	Reserved
7-5 SSW0	Snooping Strobe Width - Determines the width (in fast clock cycles) of the strobe that is outputted to IPU. Reset value is 100 - encoding 4
4–1 SWSZ0	Snooping Window Size. This field define the snooping window size as described in Table 44-20
0 SE0	Snooping Enable. This bit enables snooping detection. The M3IF monitors and detects write accesses to the snooping window. 0 Snooping feature is disabled. 1 Snooping feature is enabled.

Table 44-20. SBAR0 Field Descriptions

SWSZ0	Snooping Window Size	Window Base Address Bits	Window Address Bits in Use
0000	2 KByte	[31:11]	[10:0]
0001	4 KByte	[31:12]	[11:0]
0010	8 KByte	[31:13]	[12:0]
0011	16 KByte	[31:14]	[13:0]
0100	32 KByte	[31:15]	[14:0]
0101	64 KByte	[31:16]	[15:0]
0110	128 KByte	[31:17]	[16:0]
0111	256 KByte	[31:18]	[17:0]
1000	512 KByte	[31:19]	[18:0]
1001	1 MByte	[31:20]	[19:0]
1010	2 MByte	[31:21]	[20:0]
1011	4 MByte	[31:22]	[21:0]
1100	8 MByte	[31:23]	[22:0]
1101	16 MByte	[31:24]	[23:0]
1110	Reserved	—	—
1111	Reserved	—	—

44.2.3.16 Snooping Enable Register Low #0

The SERL0 register contains the enable bits for the lower 32 segments [31:0] in SBAR0 register.

Address 0xBASE+0x050 (SERL0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSEL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SSEL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-19. Snooping Enable Register Low #0

Details on the register fields are shown below in Table 44-21.

Table 44-21. Snooping Enable Register Low #0 Field Descriptions

Field	Description
31–0 SSEL0	Snooping Segment Enable Low 0. This register contains the enable bits for the lower 32 segments [31:0] in the snooping window (defined by the SBAR0 register). If snooping is enabled for segment #x (respective SSEL0 bit is high), than any write access detected to that segment will set the DMA_ACCESS0 for one cycle and the respective snooping status bit will be set. If the SSEL0 bit is low, and a write access to the respective segment is detected by the M3IF, only the relevant status bit in the snooping status register will be set but the DMA_ACCESS0 will not be generated. 0 Snooping segment #x is disabled. 1 Snooping segment #x is enabled.

44.2.3.17 Snooping Enable Register High #0

The SERH0 register contains the enable bits for the upper 32 segments [63:32] in SBAR0 register.

Address 0xBASE+0x054 (SERH0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSEH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SSEH0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-20. Snooping Enable Register High #0

Details on the register fields are shown below in Table 44-22.

Table 44-22. Snooping Enable Register High #0 Field Descriptions

Field	Description
31–0 SSEH0	Snooping Segment Enable High 0. This register contains the enable bits for the higher 32 segments [63:32] in the snooping window (defined by the SBAR0 register). If snooping is enabled for segment #x (respective SSEH0 bit is high), than any write access detected to that segment will set the DMA_ACCESS0 for one cycle and the respective snooping status bit will be set. If the SSEH0 bit is low, and a write access to the respective segment is detected by the M3IF, only the relevant status bit in the snooping status register will be set but the DMA_ACCESS0 will not be generated. 0 Snooping segment #x is disabled. 1 Snooping segment #x is enabled.

44.2.3.18 Snooping Status Register Low #0

The SSRL0 register contains the snooping status bits for the lower 32 segments.

Address 0xBASE+0x058 (SSRL0) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SSSL0															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSSL0															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-21. Snooping Status Register Low #0

Details on the register fields are shown below in Table 44-23.

Table 44-23. Snooping Status Register Low #0 Field Descriptions

Field	Description
31–0 SSSL0	Snooping Segment Status Low 0. This register contains the snooping status bits for the lower 32 segments [31:0] in the snooping window (defined by the SBAR0 register). A bit in the SSRL0 register is asserted if snooping to the respective segment occurred. **Note:** If snooping occurred the status bit will be updated regardless of the respective snooping segment enable bit SSEL0[x]. The DMA_ACCESS0 will be asserted only if the respective snooping segment enable bit SSEL0[x] is enabled. 0 Snooping for segment #x did not occur. 1 Snooping for segment #x occurred.

44.2.3.19 Snooping Status Register High #0

The SSRH0 register contains the snooping status bits for the higher 32 segments.

Address 0xBASE+0x05C (SSRH0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SSSH0															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSSH0															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-22. Snooping Status Register High #0

Details on the register fields are shown below in Table 44-24.

Table 44-24. Snooping Status Register High #0 Field Descriptions

Field	Description
31–0 SSSH0	Snooping Segment Status 0. This register contains the snooping status bits for the higher 32 segments [63:32] in the snooping window (defined by the SBAR0 register). A bit in the SSRH0 register is asserted if snooping to the respective segment occurred. **Note:** If snooping occurred the status bit will be updated regardless of the respective snooping segment enable bit SSEH0[x]. The DMA_ACCESS will be asserted only if the respective snooping segment enable bit SSEH0[x] is enabled. 0 Snooping for segment #x did not occur. 1 Snooping for segment #x occurred.

44.2.3.20 Snooping Base Address Register #1

The SBAR1 register contains the snooping window base address, the size of snooping window and the snooping control bit fields which are used by the fast arbitration logic to monitor the write access to CSD0, CSD1.

Address 0xBASE+0x060 (SBAR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SWBA1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SWBA1								SSW1				SWSZ1			SE1
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figure 44-23. Snooping Base Address Register #1

Details on the register fields are shown below in Table 44-25.

Table 44-25. Snooping Base Address Register #1 Field Descriptions

Field	Description
31–11 SWBA1	Snooping Window Base Address. This field defines the snooping window base address (upper 21 bits of address) to be monitored by the M3IF. M3IF monitors write accesses to the memory region above the base address window. See description in Table 44-26
10–8	Reserved
7-5 SSW1	Snooping Strobe Width - Determines the width (in fast clock cycles) of the strobe that is outputted to IPU. Reset value is 100 - encoding 4
4–1 SWSZ1	Snooping Window Size. This field define the snooping window size as described in Table 44-26
0 SE1	Snooping Enable. This bit enables snooping detection. The M3IF monitors and detects write accesses to the snooping window. 0 Snooping feature is disabled. 1 Snooping feature is enabled.

Table 44-26. SBAR1 Field Descriptions

SWSZ1	Snooping Window Size	Window Base Address Bits	Window Address Bits in Use
0000	2 KByte	[31:11]	[10:0]
0001	4 KByte	[31:12]	[11:0]
0010	8 KByte	[31:13]	[12:0]

Table 44-26. SBAR1 Field Descriptions (continued)

SWSZ1	Snooping Window Size	Window Base Address Bits	Window Address Bits in Use
0011	16 KByte	[31:14]	[13:0]
0100	32 KByte	[31:15]	[14:0]
0101	64 KByte	[31:16]	[15:0]
0110	128 KByte	[31:17]	[16:0]
0111	256 KByte	[31:18]	[17:0]
1000	512 KByte	[31:19]	[18:0]
1001	1 MByte	[31:20]	[19:0]
1010	2 MByte	[31:21]	[20:0]
1011	4 MByte	[31:22]	[21:0]
1100	8 MByte	[31:23]	[22:0]
1101	16 MByte	[31:24]	[23:0]
1110	Reserved	—	—
1111	Reserved	—	—

44.2.3.21 Snooping Enable Register Low #1

The SERL1 register contains the enable bits for the lower 32 segments [31:0] in SBAR1 register.

Address 0xBASE+0x064 (SERL1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSEL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SSEL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-24. Snooping Enable Register Low #1

Details on the register fields are shown below in Table 44-27.

Table 44-27. Snooping Base Address Register #1 Field Descriptions

Field	Description
31–0 SSEL0	Snooping Segment Enable Low 1. This register contains the enable bits for the lower 32 segments [31:0] in the snooping window (defined by the SBAR1 register). If snooping is enabled for segment #x (respective SSEL1 bit is high), than any write access detected to that segment will set the DMA_ACCESS1 for one cycle and the respective snooping status bit will be set. If the SSEL1 bit is low, and a write access to the respective segment is detected by the M3IF, only the relevant status bit in the snooping status register will be set but the DMA_ACCESS1 will not be generated. 0 Snooping segment #x is disabled. 1 Snooping segment #x is enabled.

44.2.3.22 Snooping Enable Register High #1

The SERH1 register contains the enable bits for the higher 32 segments [63:32] in SBAR1 register.

Address 0xBASE+0x068 (SERH1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SSEH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SSEH1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-25. Snooping Enable Register High #1

Details on the register fields are shown below in Table 44-28.

Table 44-28. Snooping Enable Register High #1 Field Descriptions

Field	Description
31–0 SSEH1	Snooping Segment Enable High 1. This register contains the enable bits for the higher 32 segments [63:32] in the snooping window (defined by the SBAR1 register). If snooping is enabled for segment #x (respective SSEH1 bit is high), than any write access detected to that segment will set the DMA_ACCESS1 for one cycle and the respective snooping status bit will be set. If the SSEH1 bit is low, and a write access to the respective segment is detected by the M3IF, only the relevant status bit in the snooping status register will be set but the DMA_ACCESS1 will not be generated. 0 Snooping segment #x is disabled. 1 Snooping segment #x is enabled.

44.2.3.23 Snooping Status Register Low #0

The SSRL1 register contains the snooping status bits for the lower 32 segments.

Address 0xBASE+0x06C (SSRL1) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SSSL1															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSSL1															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-26. Snooping Status Register Low #0

Details on the register fields are shown below in Table 44-29.

Table 44-29. Snooping Status Register Low #0 Field Descriptions

Field	Description
31–0 SSSL1	Snooping Segment Status Low 1. This register contains the snooping status bits for the lower 32 segments [31:0] in the snooping window (defined by the SBAR1 register). A bit in the SSRL1 register is asserted if snooping to the respective segment occurred. **Note:** If snooping occurred the status bit will be updated regardless of the respective snooping segment enable bit SSEL1[x]. The DMA_ACCESS1 will be asserted only if the respective snooping segment enable bit SSEL1[x] is enabled. 0 Snooping for segment #x did not occur. 1 Snooping for segment #x occurred.

44.2.3.24 Snooping Status Register High #1

The SSRH1 register contains the snooping status bits for the higher 32 segments.

Address 0xBASE+0x070 (SSRH1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SSSH1															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SSSH1															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-27. Snooping Status Register High #1

Details on the register fields are shown below in Table 44-30.

Table 44-30. Snooping Status Register High #1 Field Descriptions

Field	Description
31–0 SSSH1	Snooping Segment Status High 1. This register contains the snooping status bits for the higher 32 segments [63:32] in the snooping window (defined by the SBAR1 register). A bit in the SSRH1 register is asserted if snooping to the respective segment occurred. **Note:** If snooping occurred the status bit will be updated regardless of the respective snooping segment enable bit SSEH1[x]. The DMA_ACCESS1 will be asserted only if the respective snooping segment enable bit SSEH1[x] is enabled. 0 Snooping for segment #x did not occur. 1 Snooping for segment #x occurred.

44.2.3.25 I2_Unit_Level_Arbitration_ Register

The I2ULA register configure the unit level arbitration configuration bits of each logic unit in levels 3,4 and 5. Please refer to Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)" for more details on the arbitration logic units.

Address 0xBASE+0x074 (I2ULA) Access: User read-write

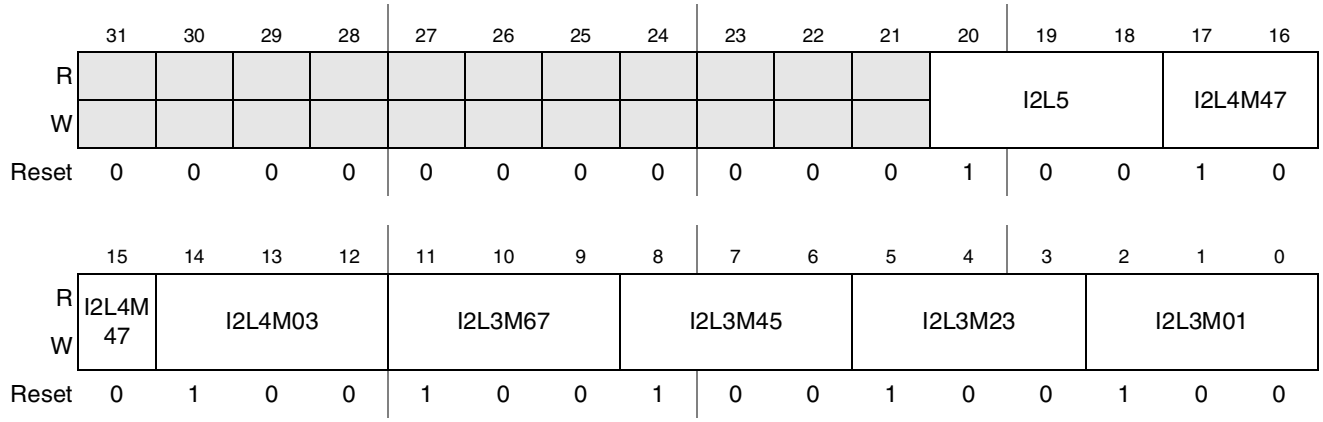


Figure 44-28. I2_Unit_Level_Arbitration_ Register

Table 44-31. Register Field Descriptions

Field	Description
31-21	Reserved.
20-18 I2L5	Int 2 Level 5 - These bits configure the arbitration mode of level 5 arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL5 is 0x100 - encoding 4.
17-15 I2L4M47	Int 2 Level 4 (m47) - These bits configure the arbitration mode of level 4(m47) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL4M47 is 0x100 - encoding 4.
14-12 I2L4M03	Int 2 Level 4 (m03) - These bits configure the arbitration mode of level 4(m03) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL4M03 is 0x100 - encoding 4.
11-9 I2L3M67	Int 2 Level 3 (m67) - These bits configure the arbitration mode of level 3(m67) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M67 is 0x100 - encoding 4.
8-6 I2L3M45	Int 2 Level 3 (m45) - These bits configure the arbitration mode of level 3(m45) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M45 is 0x100 - encoding 4.
5-3 I2L3M23	Int 2 Level 3 (m23) - These bits configure the arbitration mode of level 3(m23) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M23 is 0x100 - encoding 4.
2-0 I2L3M01	Int 2 Level 3 (m01) - These bits configure the arbitration mode of level 3(m01) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M01 is 0x100 - encoding 4.

44.2.3.26 Int. 2 Memory Arbitration Control Register

The I2ACR holds control bits for the int2. memory channel arbitration.

Address 0xBASE+0x078 (I2ACR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								I2RDT								
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Figure 44-29. Register

Table 44-32 describes the details of each field in the I2ACR register.

Table 44-32. Int. 2 Memory Arbitration Control Register Field Descriptions

Field	Description
31-9	Reserved
8 I2RDT	Read Through bit - this bit indicates the read mode from internal 2 mem. channel 0 - store and forward 1- read through (default)
7-0	Reserved

44.2.3.27 M4IF Internal 2 Control Register

The RINT2 holds control bits for the int2. memory channel arbitration.

Address 0xBASE+0x07C (RINT2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			I2DVACK	I2LPACK	I2PSS	I2PST								I2DVFS	I2LPMD	DI2PS
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1

Figure 44-30. M4IF Internal 2 Control Register

Details on the register fields are shown below in Table 44-33.

Table 44-33. M4IF Internal 2 Control Register Field Descriptions

Field	Description
31-14	Reserved
13 I2DVACK	Intr 2 DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge on the internal 2 memory channel was asserted.
12 I2LPACK	Intr 2 low-power acknowledge. This read only bit indicates whether a low-power acknowledge on the internal 2 memory channel was asserted.
11 I2PSS	Intr 2 Power Saving Status. This read only bit indicates whether internal 2 mem. channel is in power saving mode. 0 - not in power saving 1 - power saving
10-3 I2PST	Internal 2memory arbitration Power Saving Timer - these bits defines the timer value of the power saving mode of the internal 2 memory arbitration. The value in this field represent clock cycles of the internal 2 memory arbitration clock multiplied by 8. Default value is set to 80 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 8 clock cycles. ... ... 00001010 Default value- 80 clock cycles. ... ... 11111111 - timer clock is defined to 2040 clock cycles.
2 I2DVFS	Internal mem. 2 DVFS request. SW request for DVFS of internal 2 mem. channel. 0 - no dvfs request 1 - dvfs request

Table 44-33. M4IF Internal 2 Control Register Field Descriptions (continued)

Field	Description
1 I2LPMD	Internal mem. 2 LPMD request. SW request for LPMD of internal 2 mem. channel. 0 - no lpmd request 1 - lpmd request
0 DI2PS	Disable Internal 2 memory Power Saving mode - this bit when configured can disable auto power saving mode of the internal 2 memory interface. 0 - power saving is enabled 1- power saving is disabled (default mode)

44.2.3.28 Step By Step Address

Address 0xBASE+0x084 (SBS0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBS_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBS_ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-31. Register

Details on the register fields are shown below in Table 44-34

Table 44-34. Field Descriptions

Field	Description
31-0 SBS_ADDR	Step By Step Address. These bits reflect the address of a request pending on a specific arbitration/memory controller, as configured in bits [1:0] of the MDCR register.

44.2.3.29 Step By Step Address Controls

Address 0xBASE+0x088 (SBS1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R			SBS_VLD	SBS_TYPE	SBS_LEN			SBS_SIZE		SBS_BURST		SBS_PROT			SBS_LOCK	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBS_CACHE				SBS_AXI_ID				SBS_MASTER_ID				SBS_MASTER			SBS_END
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-32. Register

Details on the register fields are shown below in Table 44-35

Table 44-35. Register Field Descriptions

Field	Description
31-30 Reserved	Reserved bits.
29 SBS_VLD	Step By Step Valid. This bit reflects whether there is a pending request in the arbitration that is configured in bits [1:0] of the MDCR register. 0 - not valid 1 - valid
28 SBS_TYPE	Step By Step Request Type. These bits reflect the pending request type (read/write) as configured in bits [1:0] of the MDCR register. 0 - write 1 - read
27-25 SBS_LEN	Step By Step Length. These bits reflect the AXI “LEN” field of the pending request as configured in bits [1:0] of the MDCR register. 3’b000 - burst of length 1 3’b001 - burst of length 2 . . 3’b111 - burst of length 8
24-23 SBS_SIZE	Step By Step Size. These bits reflect the AXI “SIZE” field of the pending request as configured in bits [1:0] of the MDCR register. 2’b00 - 8 bits 2’b01 - 16 bits 2’b10 - 32 bits 2]b11 - 64 bits

Table 44-35. Register Field Descriptions (continued)

Field	Description
22-21 SBS_BURST	Step By Step Burst. These bits reflect the AXI “BURST” field of the pending request as configured in bits [1:0] of the MDCR register. 2’b00 - reserved 2’b01 - INCR burst 2’b10 - WRAP burst 2’b11 - reserved
20-18 SBS_PROT	Step By Step Protection. These bits reflect the AXI “PROT” field of the pending request as configured in bits [1:0] of the MDCR register.
17-16 SBS_LOCK	Step By Step Lock. These bits reflect the AXI “LOCK” field of the pending request as configured in bits [1:0] of the MDCR register.
15-12 SBS_CACHE	Step By Step Cache. These bits reflect the AXI “CACHE” field of the pending request as configured in bits [1:0] of the MDCR register.
11-8 SBS_AXI_ID	Step By Step AXI ID. These bits reflect the AXI “ID” field of the pending request as configured in bits [1:0] of the MDCR register.
7-4 SBS_MASTER_ID	Step By Step Master ID. These bits reflect the Master ID of the pending request as configured in bits [1:0] of the MDCR register.
3-1 SBS_MASTER	Step By Step Master. These bits reflect the EMI Master port of the pending request as configured in bits [1:0] of the MDCR register.
0 SBS_END	Step By Step Endianness. This bit reflects the endianness of the pending request as configured in bits [1:0] of the MDCR register.

44.2.3.30 M4IF Control Register #0

Address 0xBASE+0x08C (MCR0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_RST				FDVACK	SDVACK	IDVACK	DVACK	FLPACK	SLPACK	ILPACK	LPACK		FPSS	SPSS	IPSS
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EAS3	EAS2	EAS1	EAS0	FDVFS	SDVFS	IDVFS	DVFS	FLMPD	SLPMD	ILPMD	LPMD		DFPS	DSPS	DIPS
W																
Reset	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1

Figure 44-33. M4IF Control Register #0

Details on the register fields are shown below in Table 44-36.

Table 44-36. M4IF Control Register #0 Field Descriptions

Field	Description
31 SW_RST	Software reset. This bit resets all the configuration registers (except for the bit itself). 0 - reset 1 - no reset
30-28	Reserved.
27 FDVACK	Fast DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge on the fast channel was asserted.
26 SDVACK	Slow DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge on the slow channel was asserted.
25 IDVACK	Intr DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge on the internal memory channel was asserted.
24 DVACK	General DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge on the whole EMI was asserted.
23 FLPACK	Fast low-power acknowledge. This read only bit indicates whether a low-power acknowledge on the fast channel was asserted.
22 SLPACK	Slow low-power acknowledge. This read only bit indicates whether a low-power acknowledge on the slow channel was asserted.
21 ILPACK	Intr low-power acknowledge. This read only bit indicates whether a low-power acknowledge on the internal memory channel was asserted.
20 LPACK	General low-power acknowledge. This read only bit indicates whether a low-power acknowledge on the whole EMI was asserted.
19	Reserved.
18 FPSS	Fast Power Saving Status. This read only bit indicates whether fast channel is in power saving mode. 0 - not in power saving 1 - power saving
17 SPSS	Slow Power Saving Status. This read only bit indicates whether slow channel is in power saving mode. 0 - not in power saving 1 - power saving
16 IPSS	Intr Power Saving Status. This read only bit indicates whether internal mem. channel is in power saving mode. 0 - not in power saving 1 - power saving
15 EAS3	Enable Stop Arbitration M3 - Stop all read accesses to M3 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) 1- enabled (stopping) - default
14 EAS2	Enable Stop Arbitration M2 - Stop all read accesses to M2 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) - default 1- enabled (stopping)
13 EAS1	Enable Stop Arbitration M1 - Stop all read accesses to M1 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) - default 1- enabled (stopping)

Table 44-36. M4IF Control Register #0 Field Descriptions (continued)

Field	Description
12 EAS0	Enable Stop Arbitration M0 - Stop all read accesses to M0 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) 1- enabled (stopping) - default
11 FDVFS	Fast DVFS request. SW request for DVFS of fast channel. 0 - no dvfs request 1 - dvfs request
10 SDVFS	Slow DVFS request. SW request for DVFS of slow channel. 0 - no dvfs request 1 - dvfs request
9 IDVFS	Internal mem. DVFS request. SW request for DVFS of internal mem. channel. 0 - no dvfs request 1 - dvfs request
8 DVFS	General DVFS request. SW request for DVFS of whole EMI. 0 - no dvfs request 1 - dvfs request
7 FLPMD	Fast LPMD request. SW request for LPMD of fast channel. 0 - no lpmd request 1 - lpmd request
6 SLPMD	Slow LPMD request. SW request for LPMD of slow channel. 0 - no lpmd request 1 - lpmd request
5 ILPMD	Internal mem. LPMD request. SW request for LPMD of internal mem. channel. 0 - no lpmd request 1 - lpmd request
4 LPMD	General LPMD request. SW request for LPMD of whole EMI. 0 - no lpmd request 1 - lpmd request
3	Reserved.
2 DFPS	Disable Fast arb. Power Saving mode - this bit when configured can disable auto power saving mode of the fast arbitration and its memory controller. 0 - power saving is enabled 1- power saving is disabled (default mode)
1 DSPS	Disable Slow arb. Power Saving mode - this bit when configured can disable auto power saving mode of the slow arbitration and its memory controllers. 0 - power saving is enabled 1- power saving is disabled (default mode)
0 DIPS	Disable Internal memory Power Saving mode - this bit when configured can disable auto power saving mode of the internal memory interface. 0 - power saving is enabled 1- power saving is disabled (default mode)

44.2.3.31 M4IF Control Register #1

MCR1 register defines the timer values of each arbitration to be used for auto power saving mode.

Address 0xBASE+0x090 (MCR1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EAS7	EAS6	EAS5	EAS4			IPST									SPST
Reset	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SPST								FPST							
Reset	0	0	0	1	0	1	0	0	0	0	0	0	1	0	1	0

Figure 44-34. M4IF Control Register #1

Table 44-37 describes the details of each field in the MCR1 register.

Table 44-37. M4IF Control Register #1 Field Descriptions

Field	Description
31 EAS7	Enable Stop Arbtration M7 - Stop all read accesses to M7 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) 1- enabled (stopping) - default
30 EAS6	Enable Stop Arbitration M6 - Stop all read accesses to M6 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) - default 1- enabled (stopping)
29 EAS5	Enable Stop Arbitration M5 - Stop all read accesses to M5 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) - default 1- enabled (stopping)
28 EAS4	Enable Stop Arbitration M4 - Stop all read accesses to M4 that are not from to the same slave as the ones already inside the gasket. 0 - disabled (not stopping) - default 1- enabled (stopping)
27-26	Reserved

Table 44-37. M4IF Control Register #1 Field Descriptions (continued)

Field	Description
25-18 IPST	Internal 1 memory arbitration Power Saving Timer - these bits defines the timer value of the power saving mode of the internal 1 memory arbitration. The value in this field represent clock cycles of the internal 1 memory arbitration clock multiplied by 8. Default value is set to 80 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 8 clock cycles. ... ... 00001010 Default value- 80 clock cycles. ... ... 11111111 - timer clock is defined to 2040 clock cycles.
17	Reserved
16-9 SPST	Slow arbitration Power Saving Timer - these bits defines the timer value of the power saving mode of the slow arbitration. The value in this field represent clock cycles of the slow memory arbitration clock multiplied by 8. Default value is set to 80 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 8 clock cycles. ... ... 00001010 Default value- 80 clock cycles. ... ... 11111111 - timer clock is defined to 2040 clock cycles.
8	Reserved
7-0 FPST	Fast arbitration Power Saving Timer - these bits defines the timer value of the power saving mode of the fast arbitration. The value in this field represent clock cycles of the fast memory arbitration clock multiplied by 8. Default value is set to 80 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 8 clock cycles. ... ... 00001010 Default value- 80 clock cycles. ... ... 11111111 - timer clock is defined to 2040 clock cycles.

44.2.3.32 M4IF Debug Control Register

The MDCR controls several functions in the debug unit.

Address 0xBASE+0x094 (MDCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DBG_EN				DBG_RST				I2SBS_EN		SSBS_EN	DDPT	DDPM			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FSBS_EN	FSBS	SSBS		I2SBS		RARB								VARB	
W		slfclr	slfclr		slfclr											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-35. M4IF Debug Control Register

Details on the register fields are shown below in Table 44-38.

Table 44-38. M4IF Debug Control Register Field Descriptions

Field	Description
31 DBG_EN	Debug Enable. Enable debug units counters and summing. 0 - disable 1 - enable default is “disable”
30-28	Reserved.
27 DBG_RST	Debug Reset. Reset all debug unit’s counters and summing. 0 - no reset 1 - reset
26-24	Reserved.
23 I2SBS_EN	Step By Step Enable int2. Enable step by step mode for int2 channel. 0 - disable 1 - enable default is “disable”
22	Reserved.
21 I1SBS_EN	Step By Step Enable slow. Enable step by step mode for slow channel. 0 - disable 1 - enable default is “disable”

Table 44-38. M4IF Debug Control Register Field Descriptions (continued)

Field	Description
20 DDPT	Debug Dynamic Priority Type. Read or Write request reflect on the debug unit registers. 0 - write 1 - read
19-17 DDPM	Debug Dynamic Priority Master. Which master to reflect on the debug unit registers.
16	Reserved.
15 FSBS_EN	Step By Step Enable fast. Enable step by step mode for fast channel. 0 - disable 1 - enable default is “disable”
14 FSBS	Fast Step By Step. This bit trigger the fast arbitration in step by step mode. Write ‘1’ to this register in order to initiate a new transaction inside the fast arbitration. This bit goes down to ‘0’ when the new transaction is pending on AXI bus and then waiting for another write ‘1’ in order to initiate the next transaction. 1 Fast arbitration is waiting to a new transaction to go out on the AXI bus. 0 There is a pending transaction on the AXI bus.
13 SSBS	Slow Step By Step. This bit trigger the slow arbitration in step by step mode. Write ‘1’ to this register in order to initiate a new transaction inside the slow arbitration. This bit goes down to ‘0’ when the new transaction is pending on AXI bus and then waiting for another write ‘1’ in order to initiate the next transaction. 1 Slow arbitration is waiting to a new transaction to go out on the AXI bus. 0 There is a pending transaction on the AXI bus.
12 I1SBS	Int1 Step By Step. This bit trigger the int1 arbitration in step by step mode. Write ‘1’ to this register in order to initiate a new transaction inside the int1 arbitration. This bit goes down to ‘0’ when the new transaction is pending on AXI bus and then waiting for another write ‘1’ in order to initiate the next transaction. 1 Int1 arbitration is waiting to a new transaction to go out on the AXI bus. 0 There is a pending transaction on the AXI bus.
11 I2SBS	Int2 Step By Step. This bit trigger the int2 arbitration in step by step mode. Write ‘1’ to this register in order to initiate a new transaction inside the int2 arbitration. This bit goes down to ‘0’ when the new transaction is pending on AXI bus and then waiting for another write ‘1’ in order to initiate the next transaction. 1 Int2 arbitration is waiting to a new transaction to go out on the AXI bus. 0 There is a pending transaction on the AXI bus.
10	Reserved.
9-8 RARB	Read Arbitration. These 2 bits decide which arbitration will be reflected on the debug status registers when reading them. 00 - fast arbitration 01 - slow arbitration 10 - internal 1 mem. arbitration 11 - internal 2 mem. arbitration
7-2	Reserved
1-0 VARB	Visibility Arbitration. These 2 bits decide which arbitration to reflect on the EMI debug output pins and on the step by step registers. 00 - fast arbitration 01 - slow arbitration 10 - internal 1 mem. arbitration 11 - internal 2 mem. arbitration

44.2.3.33 Fast Arbitration Control Register

The FACR register is the control register of the fast arbitration logic. It contains several configuration field in order to let the user optimize the arbitration.

Address 0xBASE+0x098 (FACR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								FRDT	FDPUR							FDPE
W																
Reset	0	0	0	0	0	0	0	1	1	0	1	0	0	0	0	1

Figure 44-36. Fast Arbitration Control Register

Table 44-39 describes the details of each field in the FACR register.

Table 44-39. Fast Arbitration Control Register Field Descriptions

Field	Description
31-9	Reserved
8 FRDT	Read Through bit - this bit indicates the read mode from fast channel 0 - store and forward 1- read through (default)
7-4 FDPUR	Dynamic Priority Update Resolution - This field defines the minimum time resolution (in number of arbitration selections) that each priority master would be update by the dynamic priority logic.The minimum value is 2 selections. The value can be changed by the user in order to achieve lower resolution to reduce dynamic priority influence on the arbitration logic. It is recommended not to use higher value than average access time of the memory controller. 0000 reserved 0001 encoding 2 ... ... 1010 Default value- encoding 11. ... ... 1111 Dynamic priority calculation updates will be done once per 15 selections minimum.
3-1	Reserved
0 FDPE	Dynamic Priority Enable. This bit defines whether arbitration logic would use dynamic priority logic or not. 0 Dynamic Priority is not used, disabled. 1 Dynamic Priority is used, enabled.

44.2.3.34 F_Priority Weighting Configuration Register

The FPWC register configure the weighting values of each field in the arbitration. Each weighting field has a default value based on model simulation. It is highly recommended to use these default values and not to change it, any change to these values should be done within an extreme care.

IMPORTANT - At any configuration of these values, the user must make sure that the following restriction is kept:

FPHR+FAHR+FDPM+[highest basic priority of fast channel] < 16

Address 0xBASE+0x09C (FPWC) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R										FAHR			FPHR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R						FPCR				FPAR			FDPM			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0

Figure 44-37. F_Priority Weighting Configuration Register

Table 44-40 describes in details FPWC register fields.

/

Table 44-40. F_Priority Weighting Configuration Register Field Descriptions

Field	Description
31-23	Reserved.
22-20 FAHR	Access Hit Rate. Any pending access that is considered as page hit is checked to be a consecutive access type or not. If it is a consecutive access than the weighting value of this request will increase by FAHR value. Default value of FAHR is 0x010 - encoding 2.
19-16 FPHR	Page Hit Rate. This value will be added by the dynamic arbitration logic to any pending request that is considered as page hit. Default value of FPHR is 0x0100 - encoding 4.
15-11	Reserved.
10-8 FPCR	Pending Cycles Rate. Each timing resolution the dynamic arbitration updates a pending request, it will increase its weighting value by FPCR value. Default FPCR value is 0x001 - encoding 1.

Table 44-40. F_Priority Weighting Configuration Register Field Descriptions (continued)

Field	Description
10	Reserved.
6-5 FAPR	Access Penalty Rate. This value is decreased from any data in a served request. For example, for served access of burst2 type, the weighting value will be decreased by 2xFAPR. Default FAPR value is 0x01 - encoding 1.
4	Reserved.
3–0 FDPM	Dynamic Priority Maximum. This field defines the maximum value of the dynamic priority. This value can be changed in order to get larger window for the dynamic arbitration. Default value of FDPM is 0x1000 - encoding 8.

44.2.3.35 Slow Arbitration Control Register

The SACR register is the control register of the slow arbitration logic. It contains several configuration fields in order to let the user optimize the arbitration.

Address 0xBASE+0x0A0 (SACR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								SRDT								
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Figure 44-38. Slow Arbitration Control Register

Table 44-41 describe the details of each field in the SACR register.

Table 44-41. Slow Arbitration Control Register Field Descriptions

Field	Description
31-9	Reserved
8 SRDT	Read Through bit - this bit indicates the read mode from slow channel 0 - store and forward 1- read through (default)
7-0	Reserved

44.2.3.36 Power saving masters 2

The PSM2 determines the power saving features for masters #4 and #5.

Address 0xBASE+0x0A4 (PSM2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M5_PST									M5_WIS	M5_RIS	M5_PSS	M5_ROC			M5_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M4_PST									M4_WIS	M4_RIS	M4_PSS	M4_ROC			M4_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

Figure 44-39. Power saving masters 2 Register

Detailed description of Power saving masters 2 register is shown in Table 44-4.

Table 44-42. Power saving masters 2 Register Field Descriptions

Field	Description
31-24 M5_PST	master #5 power saving timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
23	Reserved.
22 M5_WIS	Master #5 Write Idle Status.This read only bit indicates whether master #5 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
21 M5_RIS	Master #5 Read Idle Status.This read only bit indicates whether master #5 read request buffer is idle (empty) or not. 0 - idle 1 - not idle

Table 44-42. Power saving masters 2 Register Field Descriptions (continued)

Field	Description
20 M5_PSS	Master #5 Power Saving Status. This read only bit indicates whether master #5 gasket is in power saving mode. 0 - not in power saving 1 - power saving
19-17 M5_ROC	master #5 ready off cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. 3'b001 - 3'b111 (3'b000 is a forbidden value)
16 M5_PSD	master #5 power saving disable. 0 - power saving enabled 1 - power saving disabled (default)
15-8 M4_PST	master #4 power saving timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
7	Reserved.
6 M4_WIS	Master #4 Write Idle Status.This read only bit indicates whether master #4 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
5 M4_RIS	Master #4 Read Idle Status.This read only bit indicates whether master #4 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
4 M4_PSS	Master #4 Power Saving Status. This read only bit indicates whether master #4 gasket is in power saving mode. 0 - not in power saving 1 - power saving
3-1 M4_ROC	master #4 ready off cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. Default value is 3 cycles. 3'b001 - 3'b111 (3'b000 is a forbidden value)
4 M0_PSD	master #4 power saving disable. 0 - power saving enabled 1 - power saving disabled (default)

44.2.3.37 Int. Memory Arbitration Control Register

The IACR register is the control register of the internal memory arbitration logic. It contains several configuration field in order to let the user optimize the arbitration.

Address 0xBASE+0x0A8 (IACR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								IRDT								
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Figure 44-40. Int. Memory Arbitration Control Register

Table 44-43 describes the details of each field in the IACR register.

Table 44-43. Int. Memory Arbitration Control Register Field Descriptions

Field	Description
31-9	Reserved
8 IRDT	Read Through bit - this bit indicates the read mode from internal mem. channel 0 - store and forward 1- read through (default)
7-0	Reserved

44.2.3.38 Power saving masters 3

The PSM3 determines the power saving features for masters #6 and #7.

Address 0xBASE+0x0AC (PSM3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	M7_PST									M7_WIS	M7_RIS	M7_PSS	M7_ROC			M1_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	M6_PST									M6_WIS	M6_RIS	M6_PSS	M6_ROC			M0_PSD
W																
Reset	0	0	0	0	1	0	1	0	0	0	0	0	0	1	1	1

Figure 44-41. Power saving masters 3 Register

Detailed description of Power saving masters 3 register is shown in Table 44-4.

Table 44-44. Power saving masters 3 Register Field Descriptions

Field	Description
31-24 M7_PST	master #7 power saving timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
23	Reserved.
22 M7_WIS	Master #7 Write Idle Status.This read only bit indicates whether master #7 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
21 M7_RIS	Master #7 Read Idle Status.This read only bit indicates whether master #7 read request buffer is idle (empty) or not. 0 - idle 1 - not idle

Table 44-44. Power saving masters 3 Register Field Descriptions (continued)

Field	Description
20 M7_PSS	Master #7 Power Saving Status. This read only bit indicates whether master #7 gasket is in power saving mode. 0 - not in power saving 1 - power saving
19-17 M7_ROC	master #7 ready off cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. 3'b001 - 3'b111 (3'b000 is a forbidden value)
16 M7_PSD	master #7 power saving disable. 0 - power saving enabled 1 - power saving disabled (default)
15-8 M6_PST	master #6 power saving timer. The real value which is used is register-value multiplied by 100. Default value is set to 1000 clock cycles. 00000000 Reserved - this value is forbidden. 00000001 - timer is configured to 100 clock cycles. ... ... 00001010 Default value- 1000 clock cycles. ... ... 11111111 - timer clock is defined to 25500 clock cycles.
7	Reserved.
6 M6_WIS	Master #6 Write Idle Status.This read only bit indicates whether master #6 write request buffer is idle (empty) or not. 0 - idle 1 - not idle
5 M6_RIS	Master #6 Read Idle Status.This read only bit indicates whether master #6 read request buffer is idle (empty) or not. 0 - idle 1 - not idle
4 M6_PSS	Master #6 Power Saving Status. This read only bit indicates whether master #6 gasket is in power saving mode. 0 - not in power saving 1 - power saving
3-1 M6_ROC	master #6 ready off cycles. Number of minimum cycles that the ready signals must be low before they can go back to high. Default value is 3 cycles. 3'b001 - 3'b111 (3'b000 is a forbidden value)
6 M0_PSD	master #6 power saving disable. 0 - power saving enabled 1 - power saving disabled (default)

44.2.3.39 F_Unit_Level_Arbitration_ Register

The FULA register configure the unit level arbitration configuration bits of each logic unit in levels 3,4 and 5. Please refer to Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)" for more details on the arbitration logic units.

Address 0xBASE+0x0B0 (FULA) Access: User read-write

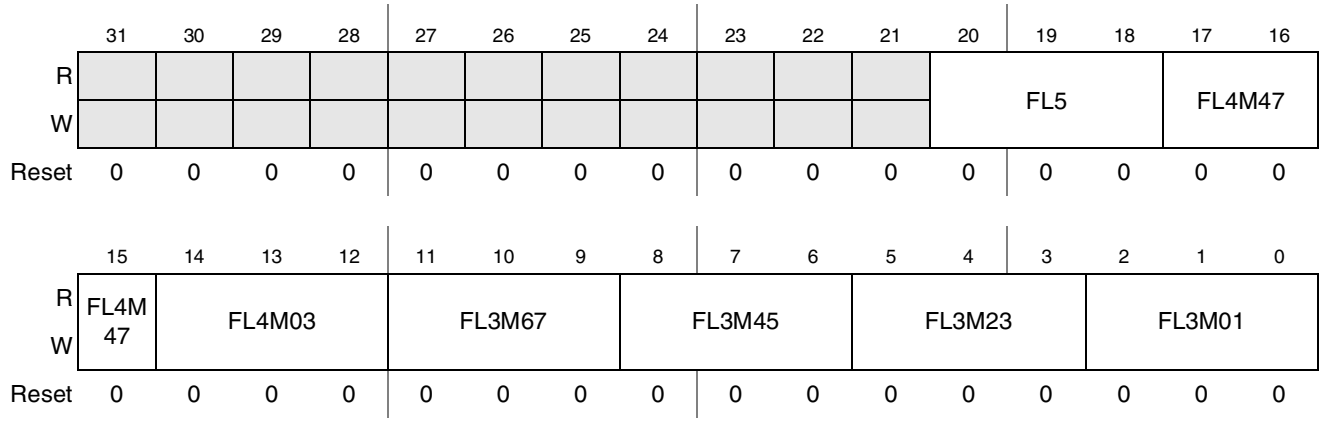


Figure 44-42. F_Unit_Level_Arbitration_ Register

Table 44-45 describes in details FULA register fields.

Table 44-45. F_Unit_Level_Arbitration_ Register Field Descriptions

Field	Description
31-21	Reserved.
20-18 FL5	Fast Level 5 - These bits configure the arbitration mode of level 5 arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL5 is 0x000 - encoding 0.
17-15 FL4M47	Fast Level 4 (m47) - These bits configure the arbitration mode of level 4(m47) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL4M47 is 0x000 - encoding 0.
14-12 FL4M03	Fast Level 4 (m03) - These bits configure the arbitration mode of level 4(m03) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL4M03 is 0x000 - encoding 0.
11-9 FL3M67	Fast Level 3 (m67) - These bits configure the arbitration mode of level 3(m67) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M67 is 0x000 - encoding 0.
8-6 FL3M45	Fast Level 3 (m45) - These bits configure the arbitration mode of level 3(m45) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M45 is 0x000 - encoding 0.
5-3 FL3M23	Fast Level 3 (m23) - These bits configure the arbitration mode of level 3(m23) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M23 is 0x000 - encoding 0.
2-0 FL3M01	Fast Level 3 (m01) - These bits configure the arbitration mode of level 3(m01) arbitration unit in the fast arbitration. The arbitration modes configuration options are shown in Table 44-62. Default value of FL3M01 is 0x000 - encoding 0.

44.2.3.40 S_Unit_Level_Arbitration_ Register

The SULA register configure the unit level arbitration configuration bits of each logic unit in levels 3,4 and 5. Please refer to Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)" for more details on the arbitration logic units.

Address 0xBASE+0x0B4 (SULA) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												SL5			SL4M47	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SL4M 47	SL4M03			SL3M67			SL3M45			SL3M23			SL3M01		
W																
Reset	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0

Figure 44-43. S_Unit_Level_Arbitration_ Register

Table 44-46 describes in details SULA register fields.

Table 44-46. S_Unit_Level_Arbitration_ Register Field Descriptions

Field	Description
31-21	Reserved.
20-18 SL5	Slow Level 5 - These bits configure the arbitration mode of level 5 arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL5 is 0x100 - encoding 4.
17-15 SL4M47	Slow Level 4 (m47) - These bits configure the arbitration mode of level 4(m47) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL4M47 is 0x100 - encoding 4.
14-12 SL4M03	Slow Level 4 (m03) - These bits configure the arbitration mode of level 4(m03) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL4M03 is 0x100 - encoding 4.
11-9 SL3M67	Slow Level 3 (m67) - These bits configure the arbitration mode of level 3(m67) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL3M67 is 0x100 - encoding 4.
8-6 SL3M45	Fast Level 3 (m45) - These bits configure the arbitration mode of level 3(m45) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL3M45 is 0x100 - encoding 4.

Table 44-46. S_Unit_Level_Arbitration_ Register Field Descriptions (continued)

Field	Description
5-3 SL3M23	Slow Level 3 (m23) - These bits configure the arbitration mode of level 3(m23) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL3M23 is 0x100 - encoding 4.
2-0 SL3M01	Slow Level 3 (m01) - These bits configure the arbitration mode of level 3(m01) arbitration unit in the slow arbitration. The arbitration modes configuration options are shown in Table . Default value of SL3M01 is 0x100 - encoding 4.

44.2.3.41 I_Unit_Level_Arbitration_ Register

The IULA register configure the unit level arbitration configuration bits of each logic unit in levels 3,4 and 5. Please refer to Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)" for more details on the arbitration logic units.

Address 0xBASE+0x0B8 (IULA) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												IL5			IL4M47	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IL4M47	IL4M03			IL3M67			IL3M45			IL3M23			IL3M01		
W																
Reset	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0

Figure 44-44. I_Unit_Level_Arbitration_ Register

Table 44-47 describes in details IULA register fields.

Table 44-47. I_Unit_Level_Arbitration_ Register Field Descriptions

Field	Description
31-21	Reserved.
20-18 IL5	Internal Mem Level 5 - These bits configure the arbitration mode of level 5 arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL5 is 0x0100 - encoding 4.
17-15 IL4M47	Internal Mem Level 4 (m47) - These bits configure the arbitration mode of level 4(m47) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL4M47 is 0x100 - encoding 4.

Table 44-47. I_Unit_Level_Arbitration_ Register Field Descriptions (continued)

Field	Description
14-12 IL4M03	Internal Mem Level 4 (m03) - These bits configure the arbitration mode of level 4(m03) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL4M03 is 0x100 - encoding 4.
11-9 IL3M67	Internal Mem Level 3 (m67) - These bits configure the arbitration mode of level 3(m67) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL3M67 is 0x100 - encoding 4.
8-6 IL3M45	Internal Mem Level 3 (m45) - These bits configure the arbitration mode of level 3(m45) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL3M45 is 0x100 - encoding 4.
5-3 IL3M23	Internal Mem Level 3 (m23) - These bits configure the arbitration mode of level 3(m23) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL3M23 is 0x100 - encoding 4.
2-0 IL3M01	Internal Mem Level 3 (m01) - These bits configure the arbitration mode of level 3(m01) arbitration unit in the Internal Mem arbitration. The arbitration modes configuration options are shown in Table . Default value of IL3M01 is 0x100 - encoding 4.

44.2.3.42 Fast_Dynamic_Priority_Status Register

The FDPS register is a read only register. This register reflect the dynamic priority value of 5 requests in the fast arbitration as configured in the FDPC register.

Address 0xBASE+0x0BC (FDPS) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R													FDP4			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FDP3				FDP2				FDP1				FDP0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-45. Fast_Dynamic_Priority_Status Register

Table 44-48 describes in details FDPS register fields.

Table 44-48. Fast_Dynamic_Priority_Status Register Field Descriptions

Field	Description
31-20	Reserved.
19-16 FDP4	Fast Dynamic Priority4 - these bits reflect the dynamic priority value of a request as configured in bits [24:20] of the FDPC register.
15-12 FDP3	Fast Dynamic Priority3 - these bits reflect the dynamic priority value of a request as configured in bits [19:15] of the FDPC register.
11-8 FDP2	Fast Dynamic Priority2 - these bits reflect the dynamic priority value of a request as configured in bits [14:10] of the FDPC register.
7-4 FDP1	Fast Dynamic Priority1 - these bits reflect the dynamic priority value of a request as configured in bits [9:5] of the FDPC register.
3-0 FDP0	Fast Dynamic Priority0 - these bits reflect the dynamic priority value of a request as configured in bits [4:0] of the FDPC register.

44.2.3.43 Fast_Dynamic_Priority_Control Register

The FDPC register is a read-write register. This register is used to configure which 5 requests' dynamic priorities will be reflected in the FDPS register.

Address 0xBASE+0x0C0 (FDPC) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R								FD4_MAS			FD4_RW		FD3_MAS			FD3_RW
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		FD2_MAS			FD2_RW		FD1_MAS			FD1_RW		FD0_MAS			FD0_RW	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-46. Fast_Dynamic_Priority_Control Register

Table 44-49 describes in details FDPC register fields.

Table 44-49. Fast_Dynamic_Priority_Control Register Field Descriptions

Field	Description
31-25	Reserved.
FD4_MAS 24-22	Fast Dynamic 4 master - These bits indicate which master is to be reflected in the FDPS register, for dynamic priority 4.
FD4_RW 21	Fast Dynamic 4 read/write - This bit indicates whether a read or a write request is to be reflected in the FDPS register, for dynamic priority 4. 0 - write 1- read
20	Reserved.
FD3_MAS 19-17	Fast Dynamic 3 master - These bits indicate which master is to be reflected in the FDPS register, for dynamic priority 3.
FD3_RW 16	Fast Dynamic 3 read/write - This bit indicates whether a read or a write request is to be reflected in the FDPS register, for dynamic priority 3. 0 - write 1- read
15	Reserved.
FD2_MAS 14-12	Fast Dynamic 2 master - These bits indicate which master is to be reflected in the FDPS register, for dynamic priority 2.
FD2_RW 11	Fast Dynamic 2 read/write - This bit indicates whether a read or a write request is to be reflected in the FDPS register, for dynamic priority 2. 0 - write 1- read
10	Reserved.
FD1_MAS 9-7	Fast Dynamic 1 master - These bits indicate which master is to be reflected in the FDPS register, for dynamic priority 1.
FD1_RW 6	Fast Dynamic 1 read/write - This bit indicates whether a read or a write request is to be reflected in the FDPS register, for dynamic priority 1. 0 - write 1- read
5	Reserved.
FD0_MAS 4-2	Fast Dynamic 0 master - These bits indicate which master is to be reflected in the FDPS register, for dynamic priority 0.
FD0_RW 1	Fast Dynamic 0 read/write - This bit indicates whether a read or a write request is to be reflected in the FDPS register, for dynamic priority 0. 0 - write 1- read
0	Reserved.

44.2.3.44 Master Len Interrupt

This is a status register that indicates whether a “LEN>8” interrupt had occurred.

Address 0xBASE+0x0C4 (MLEN) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	fb816	sb816	i1b816	i2b816					m7len	m6len	m5len	m4len	m3len	m2len	m1len	m0len
W	w1c	w1c	w1c	w1c					w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-47. Master Len Interrupt

The description of the “Master LEN Interrupt” register can be found in Table 44-50

Table 44-50. Master Len Interrupt Field Descriptions

Field	Description
31-16	Reserved
15 fb816	A burst of 16 bit or 8 bit was handled in the fast arbitration. This bit can be cleared by writing 1 to it. 0 - no burst occurred 1 - burst occurred
14 fb816	A burst of 16 bit or 8 bit was handled in the slow arbitration. This bit can be cleared by writing 1 to it. 0 - no burst occurred 1 - burst occurred
13 fb816	A burst of 16 bit or 8 bit was handled in the int1 arbitration. This bit can be cleared by writing 1 to it. 0 - no burst occurred 1 - burst occurred
12 fb816	A burst of 16 bit or 8 bit was handled in the int2 arbitration. This bit can be cleared by writing 1 to it. 0 - no burst occurred 1 - burst occurred
11-8	Reserved
m7len	Master 7 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m6len	Master 6 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.

Table 44-50. Master Len Interrupt Field Descriptions (continued)

Field	Description
m5len	Master 5 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m4len	Master 4 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m3len	Master 3 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m2len	Master 2 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m1len	Master 1 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.
m0len	Master 0 burst length value bigger then 8 status. This bit indicates if a transaction with a burst length bigger then 8 has been sent. In such case, a violation would occurred. m0len is cleared by writing one to the bit. 0 violation did not occur. 1 violation had occurred.

44.2.3.45 Watermark start ADDR_0 register

The WMSA0 register defines the watermark start address of space area #0. Watermark space #0 corresponds in general to AP privilege master which is defined by SoC AP via at the EMIv2 boundary. AP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x0D4 (WMSA0_0) Access: User read-write

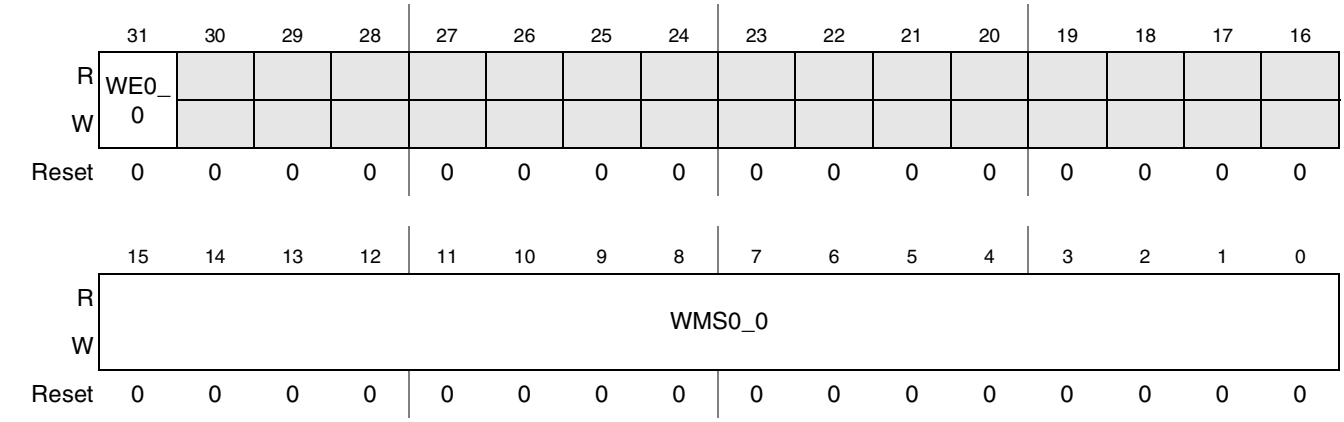


Figure 44-48. Watermark start ADDR_0 register region 0 (weim cs0)

Address 0xBASE+0x0D8 (WMSA0_1) Access: User read-write

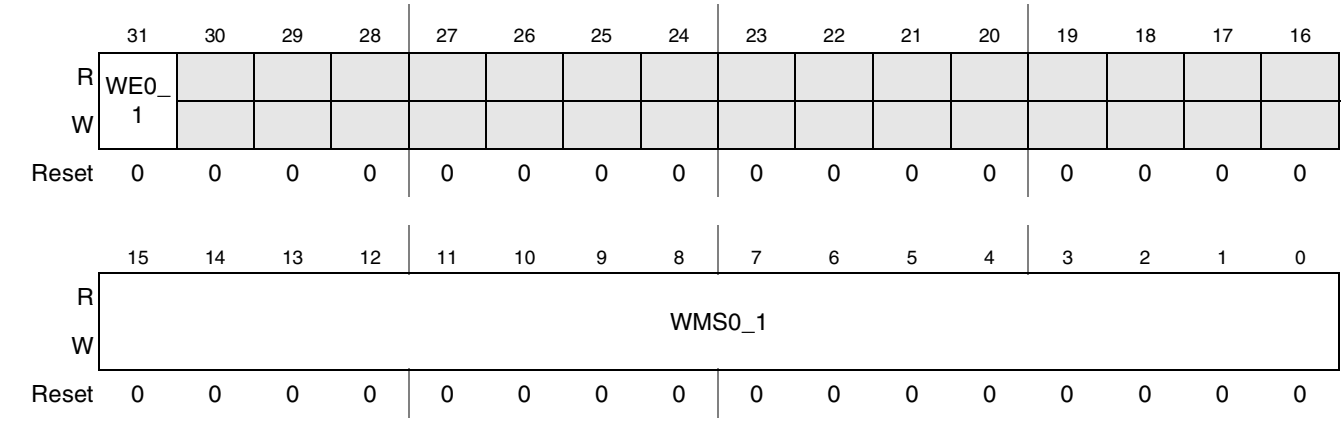


Figure 44-49. Watermark start ADDR_0 register region 1 (weim cs1)

Address 0xBASE+0x0DC (WMSA0_2) Access: User read-write

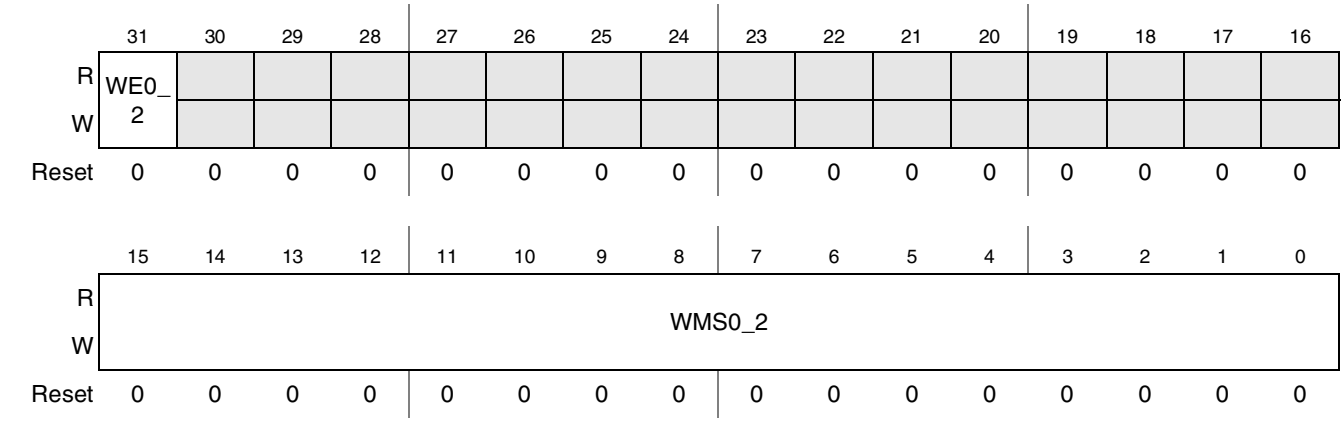


Figure 44-50. Watermark start ADDR_0 register region 2 (weim cs2)

Address 0xBASE+0x0E0 (WMSA0_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE0_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS0_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-51. Watermark start ADDR_0 register region 3 (weim cs3)

Address 0xBASE+0x0E4 (WMSA0_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE0_4															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS0_4															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-52. Watermark start ADDR_0 register region 4 (weim cs4)

Address 0xBASE+0x0E8 (WMSA0_5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE0_5															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS0_5															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-53. Watermark start ADDR_0 register region 5 (weim cs5)

Address 0xBASE+0x0EC (WMSA0_6) Access: User read-write

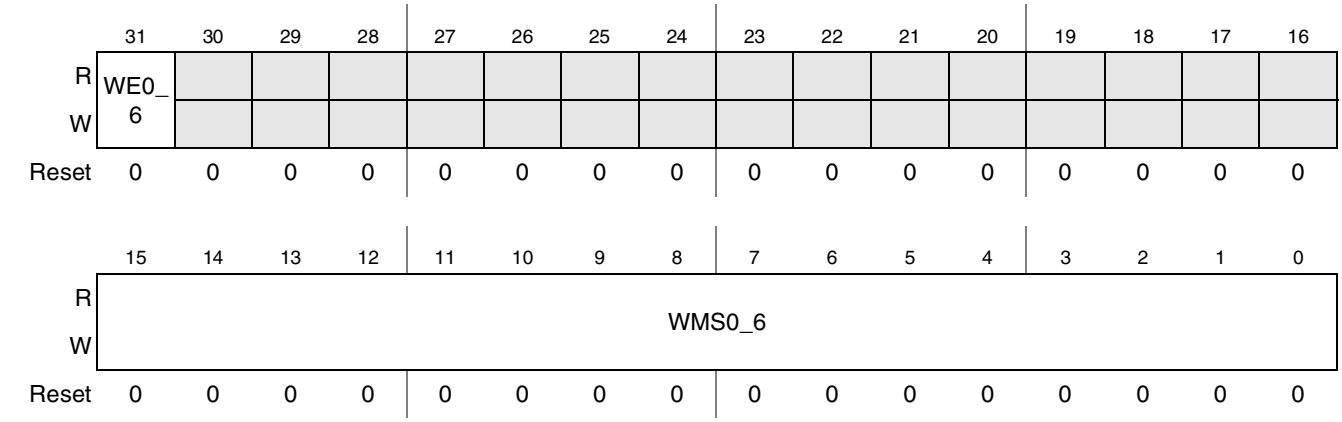


Figure 44-54. Watermark start ADDR_0 register region 6(esdctl csd0)

Address 0xBASE+0x0F0 (WMSA0_7) Access: User read-write

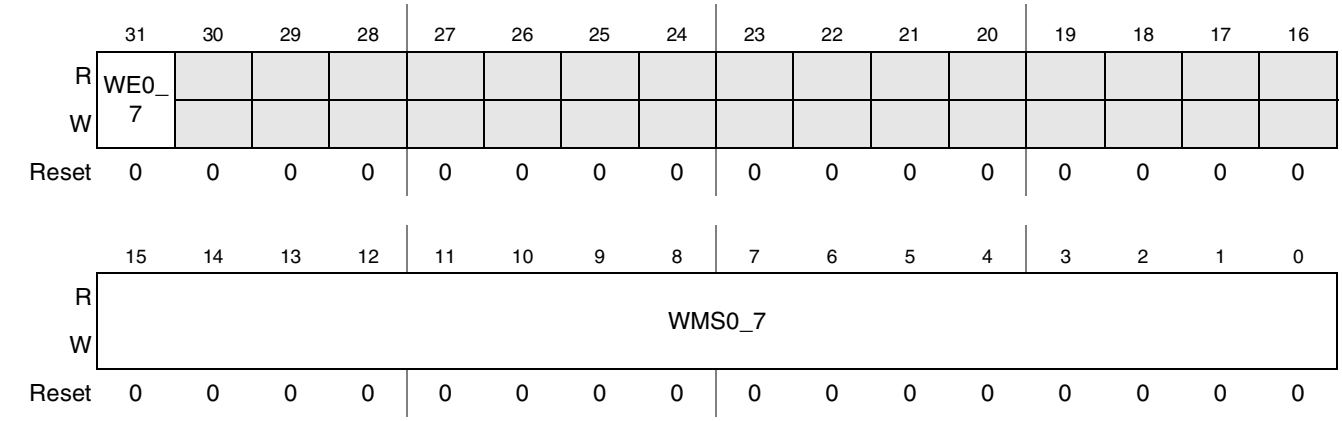


Figure 44-55. Watermark start ADDR_0 register region 7 (esdctl csd1)

Table 44-51 describes in details WMSA0_x register fields.

Table 44-51. Watermark start ADDR_0 register Field Descriptions

Field	Description
31 WE0_x	WaterMark Region Enable. This bit enables the WaterMark mechanism of space #0 Region X. 0 WaterMark Region X is disabled. 1 WaterMark Region X is enabled.
30-16	Reserved.
15-0 WMS0_x	Watermark Start Addr space #0 - As described in the watermark section in this Chapter, the watermark start address can be configured in resolution of 4KByte only. Therefore, WMSA0 register defines the upper 16bits of the start address, bits [27:12]. (bits [31:28] are determined per weim/esdctl CS) The lower 12bits that are not configurable via the register are zero always, 12'b0000_0000_0000 (0x000).

44.2.3.46 Watermark end ADDR_0 register

The WMEA0 register defines the watermark end address of space area #0. Watermark space #0 corresponds in general to AP privilege master which is defined by SoC AP via at the EMIv2 boundary. AP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x0F4 (WMEA0_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME0_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-56. Watermark end ADDR_0 register region 0 (weim cs0)

Address 0xBASE+0x0F8 (WMEA0_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME0_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-57. Watermark end ADDR_0 register region 1 (weim cs1)

Address 0xBASE+0x0FC (WMEA0_2) Access: User read-write

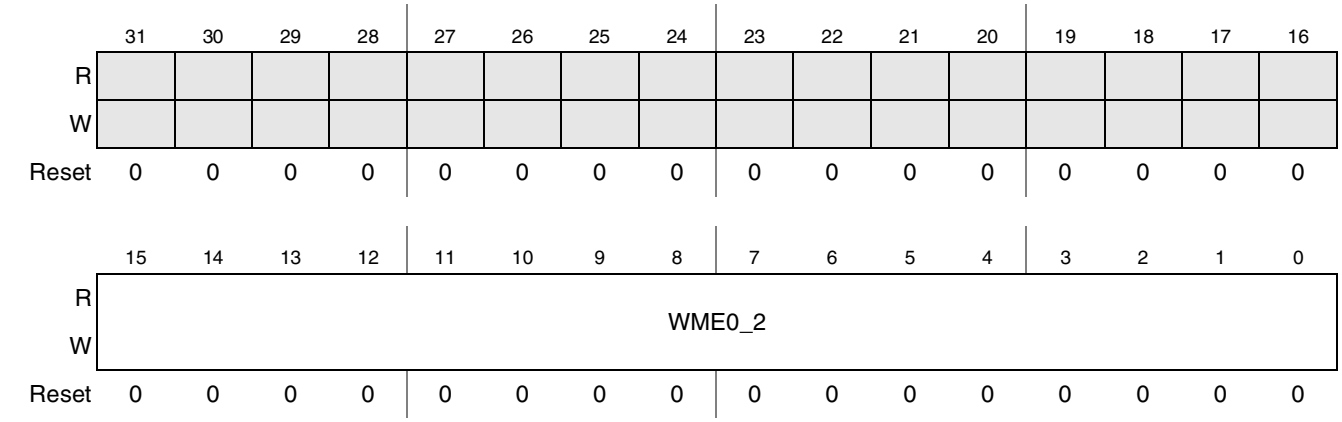


Figure 44-58. Watermark end ADDR_0 register region 2 (weim cs2)

Address 0xBASE+0x100 (WMEA0_3) Access: User read-write

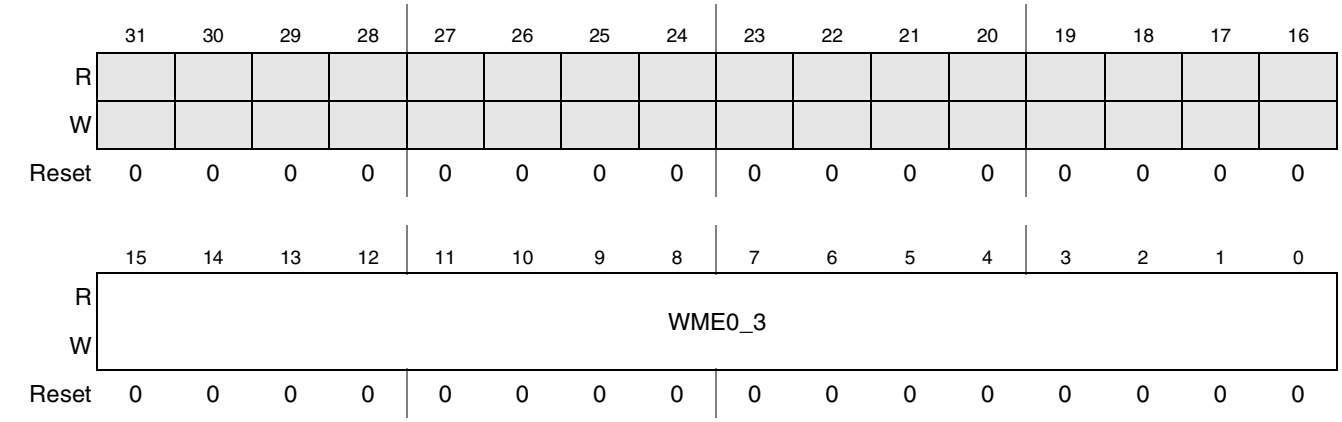


Figure 44-59. Watermark end ADDR_0 register region 3 (weim cs3)

Address 0xBASE+0x104 (WMEA0_4) Access: User read-write

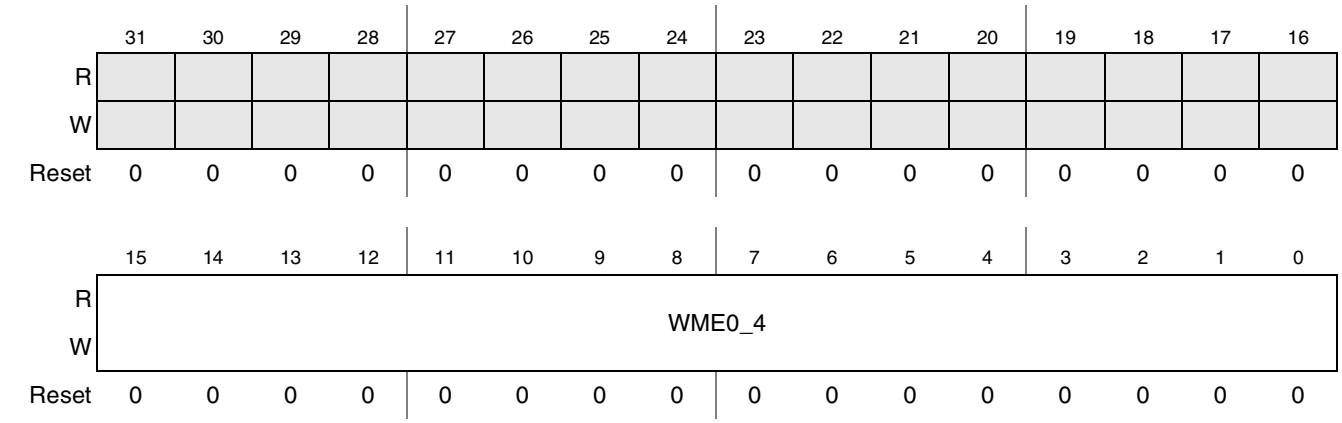


Figure 44-60. Watermark end ADDR_0 register region 4 (weim cs4)

Address 0xBASE+0x108 (WMEA0_5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME0_5															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-61. Watermark end ADDR_0 register region 5 (weim cs5)

Address 0xBASE+0x10C (WMEA0_6) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME0_6															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-62. Watermark end ADDR_0 register region 6 (esdctl csd0)

Address 0xBASE+0x110 (WMEA0_7) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME0_7															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-63. Watermark end ADDR_0 register region 7 (esdctl csd1)

Table 44-52 describes in details WMEA0_x register fields.

Table 44-52. Watermark end ADDR_0 register Field Descriptions

Field	Description
31-21	Reserved.
19-0 WME0_x	Watermark end Addr space #0 - As described in the watermark section in this Chapter, the watermark end address can be configured in resolution of 4KByte only. Therefore, WMEA0 register defines the upper 16bits of the end address, bits [27:12]. (bits [31:28] are determined per weim/esdctl CS) The lower 12bits that are not configurable via the register are one always, 12'b1111_1111_1111 (0xFFF).

44.2.3.47 Watermark Interrupt and Status #0 Register

The WMIS0 register defines the watermark interrupt enable of watermark space #0. The register enable the AP interrupt and the clear bits for each watermark space #0 region. AP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x114 (WMIS0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WIE0		FVMID_0											SVMID_0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SVMID_0								WS0_7	WS0_6	WS0_5	WS0_4	WS0_3	WS0_2	WS0_1	WS0_0
W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-64. Watermark Interrupt and Status #0 Register

Table 44-53 describes in details WMIS0 register fields.

Table 44-53. Watermark Interrupt and Status #0 Register Field Descriptions

Field	Description
31 WIE0	WaterMark Enable. This bit enables the WaterMark mechanism of space #0 including the interrupt generation. For example, if WME is set and a WaterMark violation is detected a WaterMark interrupt is generated. 0 WaterMark is disabled. 1 WaterMark is enabled.
30	Reserved.

Table 44-53. Watermark Interrupt and Status #0 Register Field Descriptions (continued)

Field	Description
29-19 FVMID0	Fast Violation Master ID - these bits contain the Master [29:27] Master ID [26:23] and AXI ID [22:19] of the master which tried to access a watermark area in the fast channel with WM[0] in low state.
18-8 SVMID0	Slow Violation Master ID - these bits contain the Master [18:16] Master ID [15:12] and AXI ID [11:8] of the master which tried to access a watermark area in the slow channel with WM[0] in low state.
7 WS0_7	WaterMark Status0_7. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of SDRAM/MDDR CS1 as defined in WMSA0_7 and WMEA0_7. WS0_7 is cleared by writing a one to the bit. 0 SDRAM/MDDR CSD1 WaterMark violation did not occur. 1 SDRAM/MDDR CSD1 WaterMark violation had occurred.
6 WS0_6	WaterMark Status0_6. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of SDRAM/MDDR CS0 as defined in WMSA0_6 and WMEA0_6. WS0_6 is cleared by writing a one to the bit. 0 SDRAM/MDDR CSD0 WaterMark violation did not occur. 1 SDRAM/MDDR CSD0 WaterMark violation had occurred.
5 WS0_5	WaterMark Status0_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS0 as defined in WMSA0_5 and WMEA0_5. WS0_5 is cleared by writing a one to the bit. 0 WEIMv2 CS5 WaterMark violation did not occur. 1 WEIMv2 CS5 WaterMark violation had occurred.
4 WS0_4	WaterMark Status0_4. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS0 as defined in WMSA0_4 and WMEA0_4. WS0_4 is cleared by writing a one to the bit. 0 WEIMv2 CS4 WaterMark violation did not occur. 1 WEIMv2 CS4 WaterMark violation had occurred.
3 WS0_3	WaterMark Status0_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS3 as defined in WMSA0_3 and WMEA0_3. WS0_3 is cleared by writing a one to the bit. 0 WEIMv2 CS3 WaterMark violation did not occur. 1 WEIMv2 CS3 WaterMark violation had occurred.
2 WS0_2	WaterMark Status0_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS0 as defined in WMSA0_2 and WMEA0_2. WS0_2 is cleared by writing a one to the bit. 0 WEIMv2 CS2 WaterMark violation did not occur. 1 WEIMv2 CS2 WaterMark violation had occurred.
1 WS0_1	WaterMark Status0_1. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS0 as defined in WMSA0_1 and WMEA0_1. WS0_1 is cleared by writing a one to the bit. 0 WEIMv2 CS1 WaterMark violation did not occur. 1 WEIMv2 CS1 WaterMark violation had occurred.
0 WS0_0	WaterMark Status0_0. This bit indicates if WaterMark violation had occurred in the WaterMark space #0 memory region of WEIMv2 CS0 as defined in WMSA0_0 and WMEA0_0. WS0_0is cleared by writing a one to the bit. 0 WEIMv2 CS0 WaterMark violation did not occur. 1 WEIMv2 CS0 WaterMark violation had occurred.

44.2.3.48 Fast Watermark Violation Address #0 register

The FWMVA0 register contains the address of the access that violated the watermark space #0 for fast channel (SDR/DDR) corresponding to watermark interrupt and status #0 register. AP privilege master is the only master that can read this register.

Address 0xBASE+0x118 (FWMVA0) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FWAVA0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FWAVA0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-65. Fast Watermark Violation Address #0 register

Table 44-54 describes in details FWMVA0 register fields.

Table 44-54. Fast Watermark Violation Address #0 register Field Descriptions

Field	Description
31-0 FWAVA0	Fast WaterMark Access violation address - These bits reflect the address of the access that caused the watermark space #0 violation in fast channel (SDR/DDR).

44.2.3.49 Slow Watermark Violation Address #0 register

The SWMVA0 register contains the address of the access that violated the watermark space #0 for slow channel (WEIM) corresponding to watermark interrupt and status #0 register. AP privilege master is the only master that can read this register.

Address 0xBASE+0x11C (SWMVA0) Access: User read only

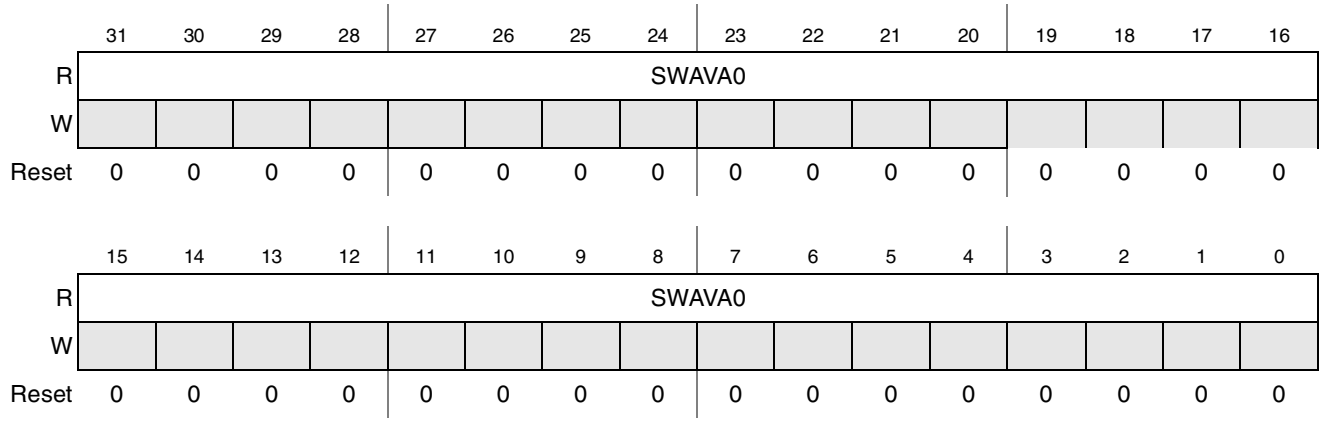


Figure 44-66. Slow Watermark Violation Address #0 register

Table 44-55 describes in details SWMVA0 register fields.

Table 44-55. Slow Watermark Violation Address #0 register Field Descriptions

Field	Description
31-0 SWAVA0	Slow WaterMark Access violation address - These bits reflect the address of the access that caused the watermark space #0 violation in slow channel (WEIM).

44.2.3.50 Watermark start ADDR_1 Register

The WMSA1 register defines the watermark start address of space area #1. Watermark space #1 corresponds in general to BP privilege master which is defined by SoC BP via at the EMIv2 boundary. BP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x120 (WMSA1_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE1_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS1_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-67. Watermark start ADDR_1 Register region 0 (weim cs0)

Address 0xBASE+0x124 (WMSA1_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE1_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS1_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-68. Watermark start ADDR_1 Register region 1(weim cs1)

Address 0xBASE+0x128 (WMSA1_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WE1_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WMS1_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-69. Watermark start ADDR_1 Register region 2 (weim cs2)

Address 0xBASE+0x12C (WMSA1_3) Access: User read-write

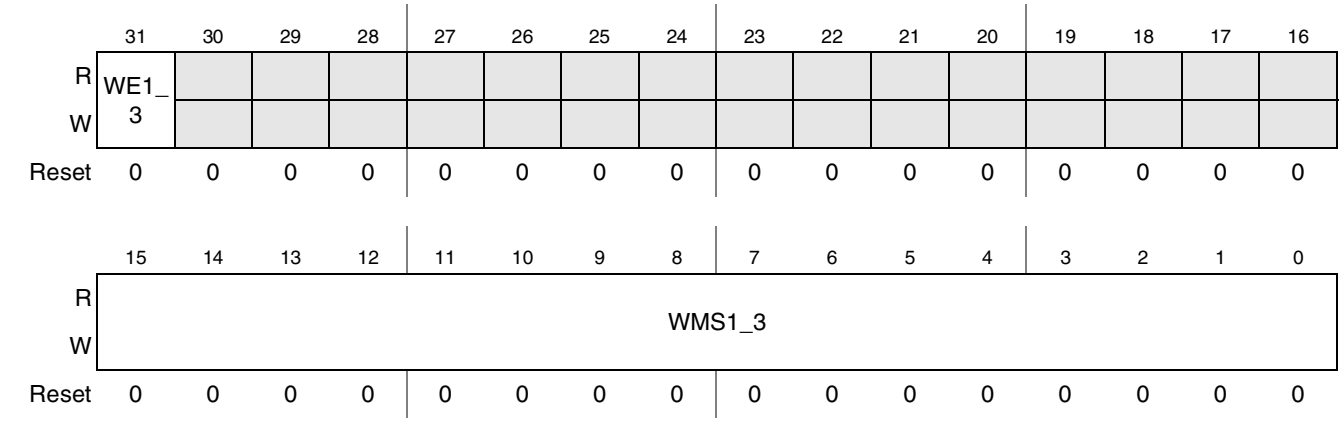


Figure 44-70. Watermark start ADDR_1 Register region 3 (weim cs3)

Address 0xBASE+0x130 (WMSA1_4) Access: User read-write

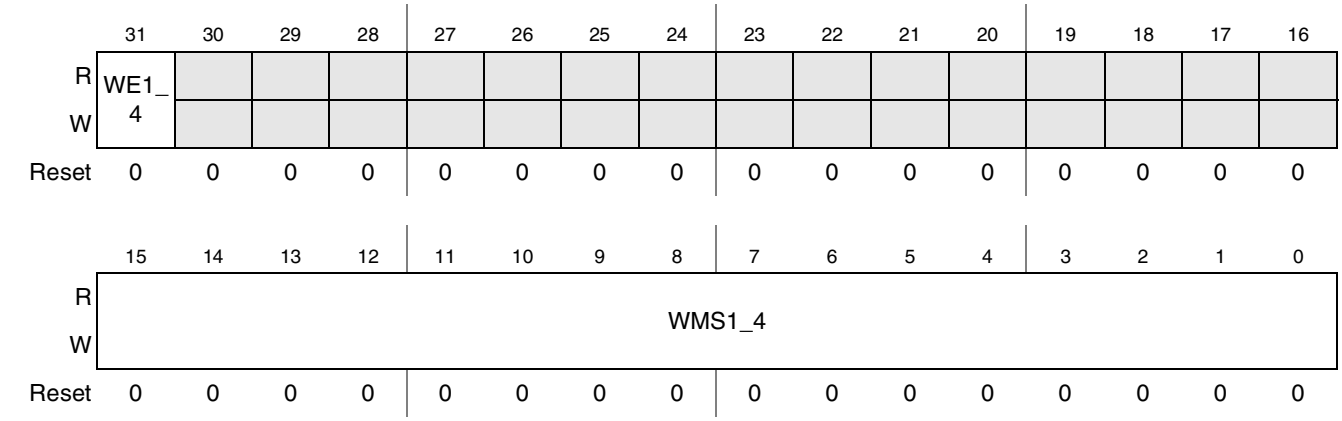


Figure 44-71. Watermark start ADDR_1 Register region 4 (weim cs4)

Address 0xBASE+0x134 (WMSA1_5) Access: User read-write

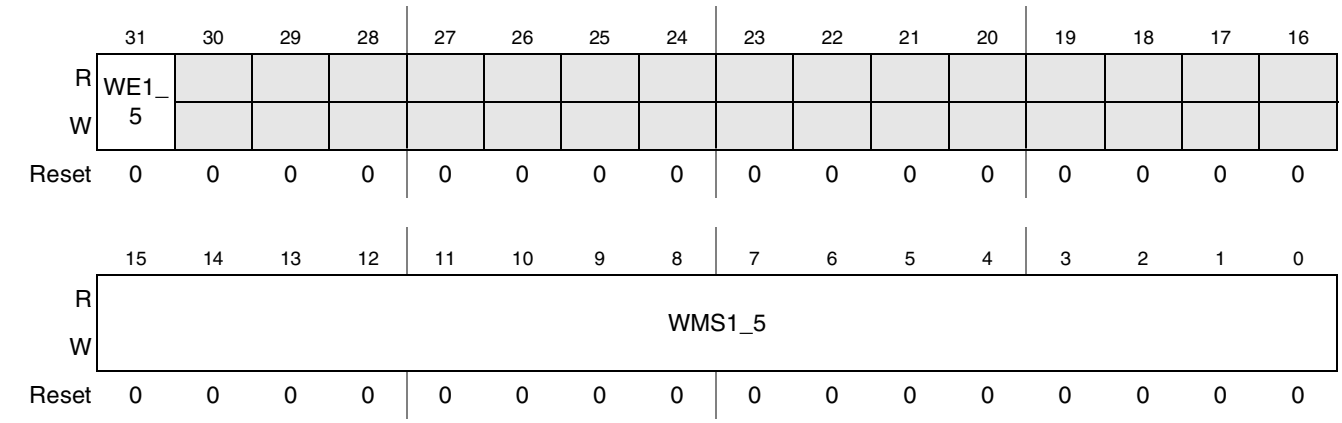


Figure 44-72. Watermark start ADDR_1 Register region 5 (weim cs5)

Address 0xBASE+0x138 (WMSA1_6) Access: User read-write

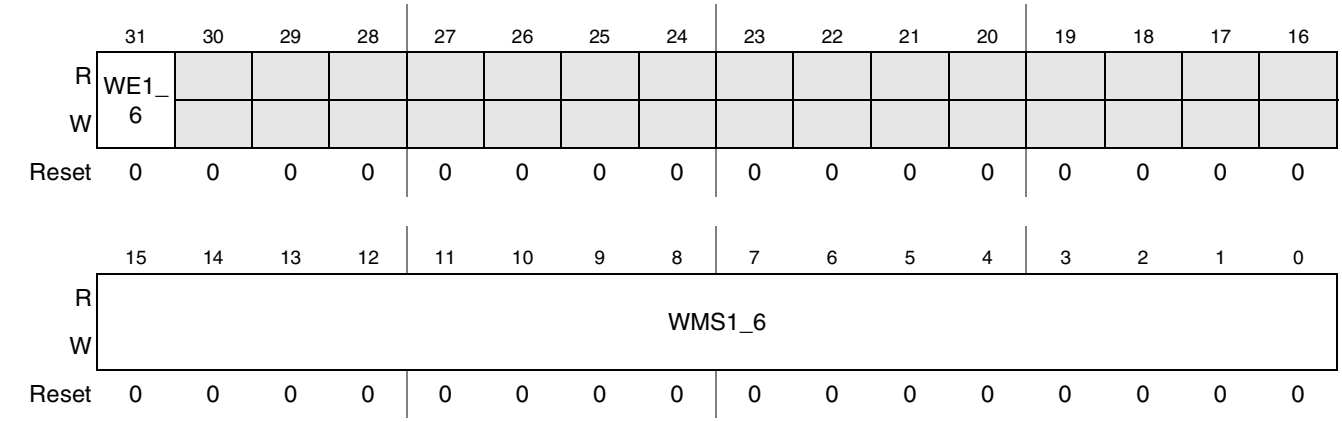


Figure 44-73. Watermark start ADDR_1 Register region 6 (esdctl csd0)

Address 0xBASE+0x13C (WMSA1_7) Access: User read-write

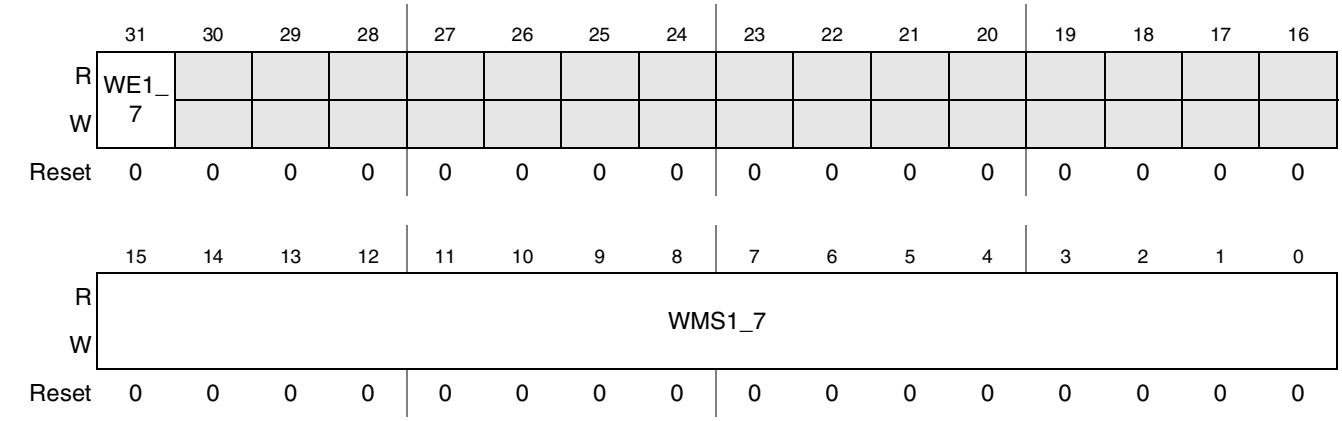


Figure 44-74. Watermark start ADDR_1 Register region 7(esdctl csd1)

Table 44-56 describes in details WMSA1_x register fields.

Table 44-56. Watermark start ADDR_1 Register Field Descriptions

Field	Description
31 WE1_x	WaterMark Region Enable. This bit enables the WaterMark mechanism of space #1 Region X. 0 WaterMark Region X is disabled. 1 WaterMark Region X is enabled.
30-21	Reserved.
20-0 WMS1_x	Watermark Start Addr space #1 - As described in the watermark section in this Chapter, the watermark start address can be configured in resolution of 4KByte only. Therefore, WMSA0 register defines the upper 16bits of the start address, bits [27:12]. (bits [31:28] are determined per weim/esdctl CS) The lower 12bits that are not configurable via this register are zero always, 12'b0000_0000_0000 (0x000).

44.2.3.51 Watermark end ADDR_1 Register

The WMEA1 register defines the watermark end address of space area #1. Watermark space #1 corresponds in general to BP privilege master which is defined by SoC BP via at the EMIv2 boundary. BP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x140 (WMEA1_0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME1_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-75. Watermark end ADDR_1 Register region 0 (weim cs0)

Address 0xBASE+0x144 (WMEA1_1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME1_1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-76. Watermark end ADDR_1 Register region 1 (weim cs1)

Address 0xBASE+0x148 (WMEA1_2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME1_2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-77. Watermark end ADDR_1 Register region 2 (weim cs2)

Address 0xBASE+0x14C (WMEA1_3) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME1_3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-78. Watermark end ADDR_1 Register region 3 (weim cs3)

Address 0xBASE+0x150 (WMEA1_4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WME1_4															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-79. Watermark end ADDR_1 Register region 4 (weim cs4)

Address 0xBASE+0x154 (WMEA1_5) Access: User read-write

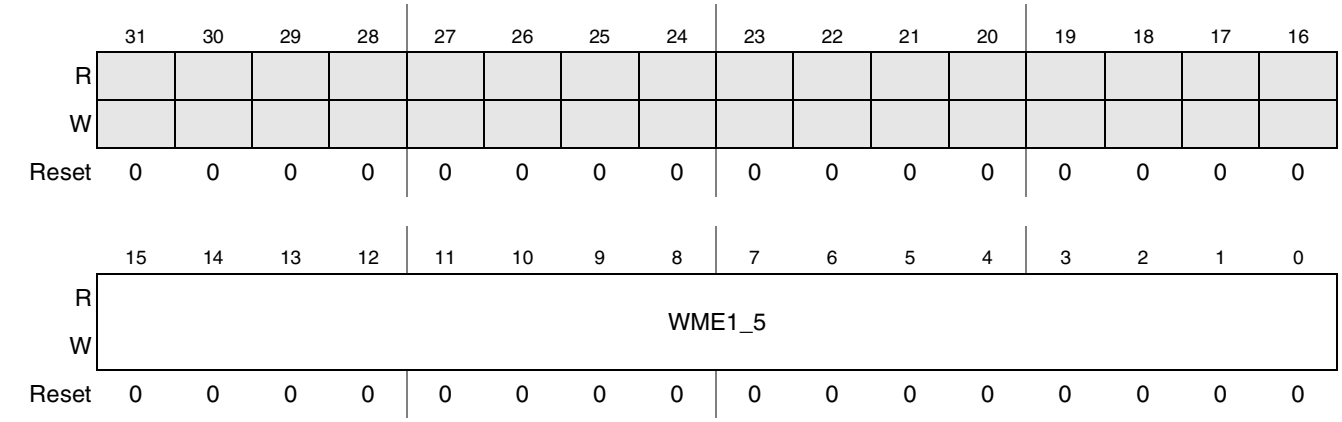


Figure 44-80. Watermark end ADDR_1 Register region 5 (weim cs5)

Address 0xBASE+0x158 (WMEA1_6) Access: User read-write

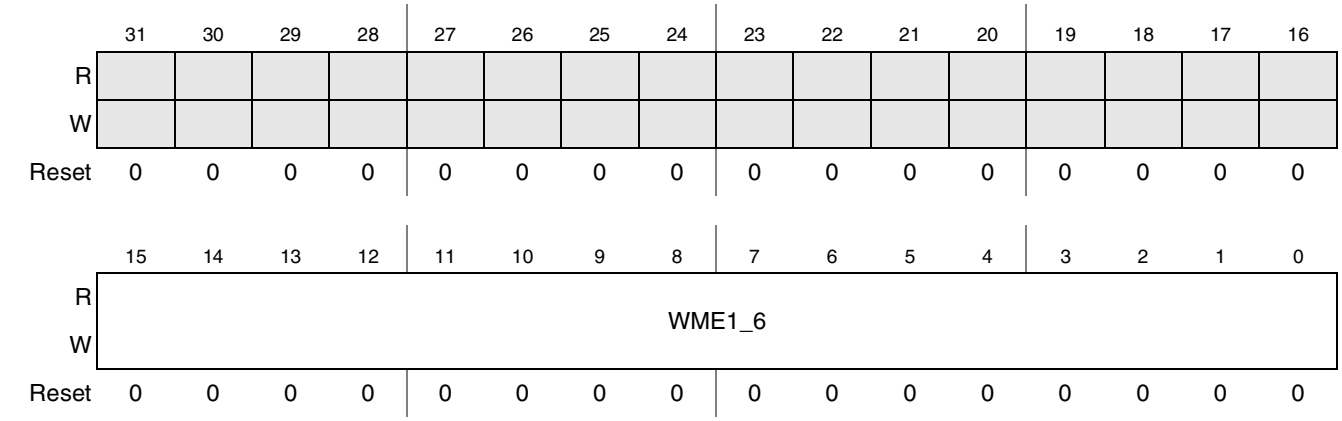


Figure 44-81. Watermark end ADDR_1 Register region 6 (esdctl csd0)

Address 0xBASE+0x15C (WMEA1_7) Access: User read-write

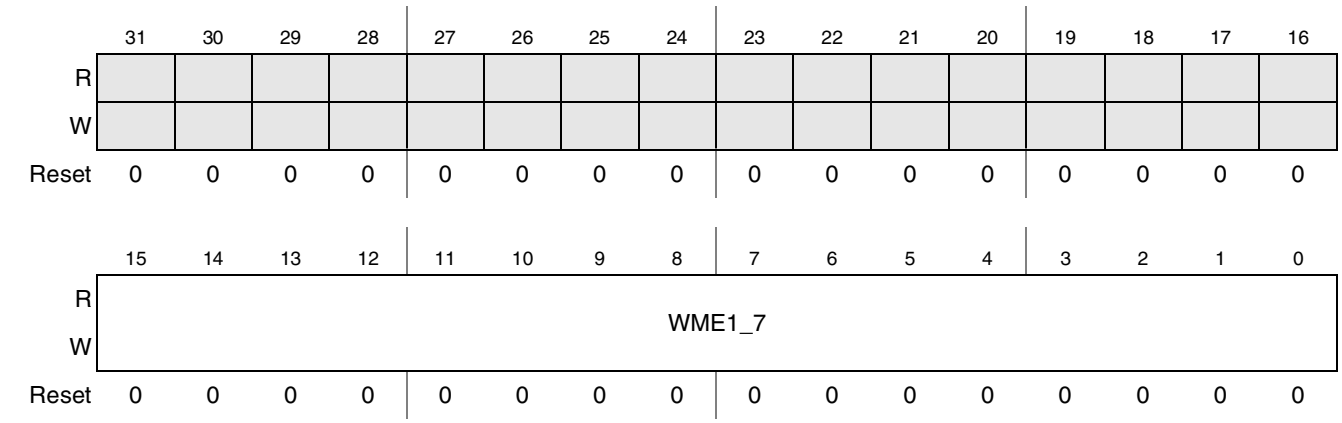


Figure 44-82. Watermark end ADDR_1 Register region 7 (esdctl csd1)

Table 44-57 describes in details WMEA1_x register fields.

Table 44-57. Watermark end ADDR_1 Register Field Descriptions

Field	Description
31-21	Reserved.
15-0 WME1_x	Watermark end Addr space #1 - As described in the watermark section in this Chapter, the watermark end address can be configured in resolution of 4KByte only. Therefore, WMEA0 register defines the upper 16bits of the end address, bits [27:12]. (bits [31:28] are determined per weim/esdctl CS) The lower 12bits that are not configurable via the register are one always, 12'b1111_1111_1111 (0xFFF).

Table 44-58 map watermark registers to their corresponding CS in EMIv2.

Table 44-58. Watermark Register Mapping

Watermark registers	CS #	Module name
WMSA0_0 WMEA0_0 WMSA1_0 WMEA1_0	CS0	WEIMv2
WMSA0_1 WMEA0_1 WMSA1_1 WMEA1_1	CS1	WEIMv2
WMSA0_2 WMEA0_2 WMSA1_2 WMEA1_2	CS2	WEIMv2
WMSA0_3 WMEA0_3 WMSA1_3 WMEA1_3	CS3	WEIMv2
WMSA0_4 WMEA0_4 WMSA1_4 WMEA1_4	CS4	WEIMv2
WMSA0_5 WMEA0_5 WMSA1_5 WMEA1_5	CS5	WEIMv2
WMSA0_6 WMEA0_6 WMSA1_6 WMEA1_6	CSD0	ESDCTLv2
WMSA0_7 WMEA0_7 WMSA1_7 WMEA1_7	CSD1	ESDCTLv2

Table 44-58. Watermark Register Mapping (continued)

Watermark registers	CS #	Module name
Watermark is not supported		NFCv2
Watermark is not supported		Internal Memory

NOTE: If a watermark register is configured to addresses which are not included in the CS it belongs to, unknown behavior will occur.

44.2.3.52 Watermark Interrupt and Status #1 Register

The WMIS1 register defines the watermark interrupt enable of watermark space #1. The register enable the BP interrupt and the clear bits for each watermark space #1 region. BP privilege master is the only master that can configure/read this register.

Address 0xBASE+0x160 (WMIS1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	WIE1		FVMID_1											SVMID_1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SVMID_1								WS1_7	WS1_6	WS1_5	WS1_4	WS1_3	WS1_2	WS1_1	WS1_0
W									w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-83. Watermark Interrupt and Status #1 Register

Table 44-59 describes in details WMIS1 register fields.

Table 44-59. Watermark Interrupt and Status #1 Register Field Descriptions

Field	Description
31 WIE1	WaterMark Enable. This bit enables the WaterMark mechanism of space #1 including the interrupt generation. For example, if WIE is set and a WaterMark violation is detected a WaterMark interrupt is generated. 0 WaterMark is disabled. 1 WaterMark is enabled.
30	Reserved.
29-19 FVMID1	Fast Violation Master ID - these bits contain the Master [29:27] Master ID [26:23] and AXI ID [22:19] of the master which tried to access a watermark area in the fast channel with WM[1] in low state.

Table 44-59. Watermark Interrupt and Status #1 Register Field Descriptions (continued)

Field	Description
18-8 SVMID1	Slow Violation Master ID - these bits contain the Master [18:16] Master ID [15:12] and AXI ID [11:8] of the master which tried to access a watermark area in the slow channel with WM[1] in low state.
7 WS1_7	WaterMark Status1_7. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of SDRAM/MDDR CS1 as defined in WMSA1_7 and WMEA1_7. WS1_7 is cleared by writing a one to the bit. 0 SDRAM/MDDR CSD1 WaterMark violation did not occur. 1 SDRAM/MDDR CSD1 WaterMark violation had occurred.
6 WS1_6	WaterMark Status1_6. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of SDRAM/MDDR CS0 as defined in WMSA1_6 and WMEA1_6. WS1_6 is cleared by writing a one to the bit. 0 SDRAM/MDDR CSD0 WaterMark violation did not occur. 1 SDRAM/MDDR CSD0 WaterMark violation had occurred.
5 WS1_5	WaterMark Status1_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS0 as defined in WMSA1_5 and WMEA1_5. WS1_5 is cleared by writing a one to the bit. 0 WEIMv2 CS5 WaterMark violation did not occur. 1 WEIMv2 CS5 WaterMark violation had occurred.
4 WS1_4	WaterMark Status1_4. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS0 as defined in WMSA1_4 and WMEA1_4. WS1_4 is cleared by writing a one to the bit. 0 WEIMv2 CS4 WaterMark violation did not occur. 1 WEIMv2 CS4 WaterMark violation had occurred.
3 WS1_3	WaterMark Status1_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS3 as defined in WMSA1_3 and WMEA1_3. WS1_3 is cleared by writing a one to the bit. 0 WEIMv2 CS3 WaterMark violation did not occur. 1 WEIMv2 CS3 WaterMark violation had occurred.
2 WS1_2	WaterMark Status1_5. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS0 as defined in WMSA1_2 and WMEA1_2. WS1_2 is cleared by writing a one to the bit. 0 WEIMv2 CS2 WaterMark violation did not occur. 1 WEIMv2 CS2 WaterMark violation had occurred.
1 WS1_1	WaterMark Status1_1. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS0 as defined in WMSA1_1 and WMEA1_1. WS1_1 is cleared by writing a one to the bit. 0 WEIMv2 CS1 WaterMark violation did not occur. 1 WEIMv2 CS1 WaterMark violation had occurred.
0 WS1_0	WaterMark Status1_0. This bit indicates if WaterMark violation had occurred in the WaterMark space #1 memory region of WEIMv2 CS0 as defined in WMSA1_0 and WMEA1_0. WS1_0is cleared by writing a one to the bit. 0 WEIMv2 CS0 WaterMark violation did not occur. 1 WEIMv2 CS0 WaterMark violation had occurred.

44.2.3.53 Fast Watermark Violation Address #1 Register

The FWMVA0 register contains the address of the access that violated the watermark space #1 for fast channel (SDR/DDR) corresponding to watermark interrupt and status #1 register. BP privilege master is the only master that can read this register.

Address 0xBASE+0x164 (FWMVA1) Access: User read only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FWAVA1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FWAVA1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 44-84. Fast Watermark Violation Address #1 Register

Table 44-60 describes in details FWMVA1 register fields.

Table 44-60. Fast Watermark Violation Address #1 Register Field Descriptions

Field	Description
31-0 FWAVA1	Fast WaterMark Access violation address - These bits reflect the address of the access that caused the watermark space #1 violation in fast channel (SDR/DDR).

44.2.3.54 Slow Watermark Violation Address #1 Register

The SWMVA0 register contains the address of the access that violated the watermark space #1 for slow channel (WEIM) corresponding to watermark interrupt and status #1 register. BP privilege master is the only master that can read this register.

Address 0xBASE+0x168 (SWMVA1) Access: User read only

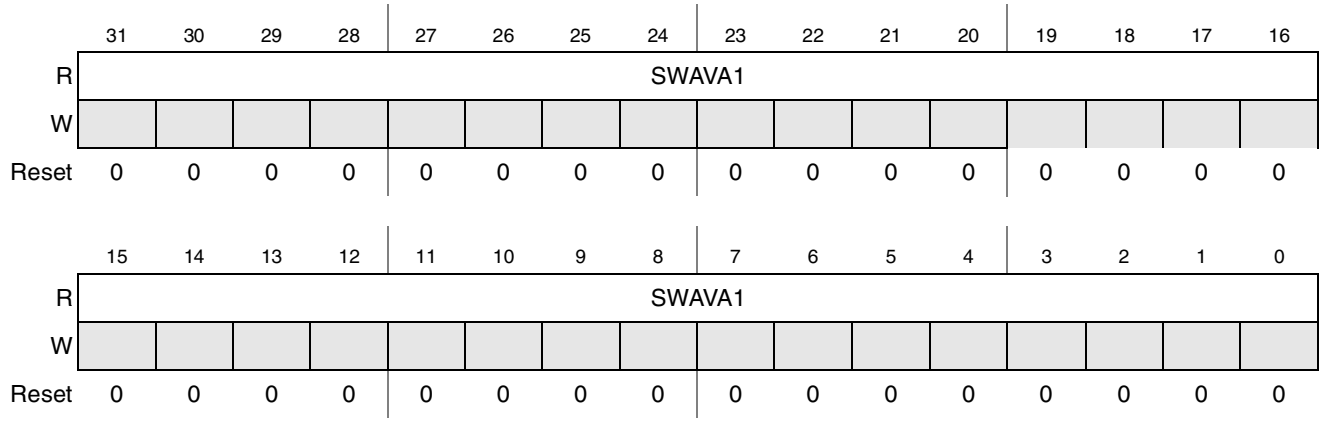


Figure 44-85. Slow Watermark Violation Address #1 Register

Table 44-61 describes in details SWMVA1 register fields.

Table 44-61. Slow Watermark Violation Address #1 Register Field Descriptions

Field	Description
31-0 SWAVA1	Slow WaterMark Access violation address - These bits reflect the address of the access that caused the watermark space #1 violation in slow channel (WEIM).

44.3 Functional Description

This section and the sub section below provide a functional description on the internal logic of M4IF. This section describes in high level the way an access paths between system level through M4IF till the memory controller. The sub sections below describes in more details each part of M4IF sub modules and its functionality.

44.3.1 Write Access Description

An AXI access is detected on one or more AXI port gaskets and latched into the addr/ctrl internal buffer. If the buffer is not full, the access can be served and data starts to enter to the write buffer. As was shown in Figure 44.1.2 the data is latched in the master clock domain. When WLAST signal is detected on the bus, an internal logic inside the gasket generates a request to the right arbitration module based on the address decoding that was done when the addr first arrived to the gasket. Since the request is generated in the master clock domain it is needed to sync it to the arbitration clock domain. This way it is guaranteed by design that all the data is stored in the buffer when the request is high at the arbitration.

The arbitration module starts to calculate priority to this new valid request and chooses it according to the arbitration's criterion. At the moment the access was chosen by the arbitration, it passes to the memory controller. The memory controller gets the data from the write buffer based on the **one_hot_wr_sel** control

of the buffer that is increment in the memory controller's clock domain. When the last data is pulled out of the buffer to the memory controller, the arbitration generates the WLAST signal as required in AXI bus protocol. The spot in the write buffer (and in the gasket's addr/ctrl buffer) will be cleared when a valid write response is issued from the gasket to the master.

In terms of write response, the gasket sends a **bufferable** response on the AXI response channel the moment the last data has reached the specific memory controller. In case of a **non-bufferable** access the response would be sent only after the write access is finished in the memory controller, and the memory controller itself sent a response.

44.3.2 Read Access Description

In a similar way when the access is a read access, the AXI addr and control signals are latched in a read control buffer inside the gasket. They are then transferred to the right arbitration module according to the address decoding in the gasket. Based on the arbitration scheme the request is served. The addr and control signals are sent to the memory controller. Data that comes out of the controller is written directly to the read buffer. The read buffer is dedicated to the specific memory controller, and is shared between all 8 AXI masters. The master can pull the data out of the buffer at the moment it gets RVALID signal from the memory controller.

A detailed description of M4IF sub module's functionality is described in the following sub sections.

44.3.3 AXI Port Gasket Functional Description

The AXI port gasket is an AXI interface to any master in the system. The gasket is a x64 AXI bus interface and can be fitted to a x32 AXI interface as well. In order to connect a x32 master to the gasket, the wdata and wstrb bus should be duplicated. The rdata bus should connect to the lower 32 bits out of the 64 that come out of the read buffer.

The gasket works with Data Interleaving Depth (DID) of 1. This is a big difference from previous versions of M4IF, where the gasket worked with DID of 2. It can hold as many accesses as its buffer depth (which is parametric) regardless of the AXI ID. This is another noticeable change from the previous M4IF, which was limited to 2 pending IDs.

Each gasket can generate up to two parallel requests to the arbitration logic, one for write and one for read. All requests can be valid in parallel to the same or to different arbitration modules.

The read addr fifo and read requests logic is similar to the write logic. The same fifo exist also for other AXI control signal so it can give the whole information to the arbitration whenever a request is valid at the arbitration module.

The write data is coming on the bus from the master directly into the write buffer. Since the write buffer is dedicated to each master that data is latched in the same clock domain of the master and no synchronization is needed.

44.3.4 Read/Write Buffer Functional Description

M4IF module includes up to 12 data buffers. Up to 8 write data buffers, and 4 read data buffers. The write buffers are dedicated for each master in the system. If a master is removed from the system, its gasket and its write buffer are removed, thus saving a lot of area. The read data buffers are dedicated for each arbitration. In the read buffer data is written by the specific controller and pulled out by any of the 8 masters, while in the write buffer, the dedicated master writes the data into the buffer and each of the 4 arbitration modules can pull out the data.

Therefore, in the write buffer the data is written in the master clock domain while in the read buffer the data is written in the specific arbitration clock domain.

Both the read and write buffer types, divided into fixed data slots in which data can be written. The slots are of depth 8 (the maximal burst length) and of 64bits size. In addition to the data, the buffers store the write strobe (8 bit) for write, and the read response (rresp, 2bit) for read. Each slot has a pointer_in and a pointer_out to mange the two way data, the way in which it is written into the slot and the way it pulled out of the slot. It is important to mention that data of a certain access can not be stored in two different slots.

There are two configuration options to pull the data out of the read buffers, “read through” and “store & forward”. In “store and forward” operation mode, the data can be pulled out by the slave only when all the datum of the burst is stored in the buffer, that is,. after RVALID, RREADY and RLAST signals are high on the bus. In “read trough” operation mode, the data can be pulled out of the buffer by the slave a few cycles after the first data was written to buffer, due to synchronization reasons.
For fast memory controllers it is recommended to use the “read through” and therefore to achieve a minimum latency between the controller and the master. However, for slow memory controller when there is an option that it would take time for the memory controller to write the data into the buffer, it is recommended to use the “store & forward” operation mode and to ensure the master bus won’t be busy for a long period of time.

On the write data buffer it is simpler, there is only one operation mode, the “store & forward” mode. In order to achieve best performance the arbitration guarantees that when a request is served and moves to the memory controller, all the data is already latched in the write buffer and can be pulled out by the memory controller without any busy or abort cycles in between each data word of the access. This happens since the access request is asserted at the arbitration only after the last data was written to the buffer.

The write buffer gets data from its master. Data can be pulled out of the write buffers by the 4 memory controllers in parallel. In the read buffer, only one memory controller can write to the buffer but several masters can pull the data at the same time, depending on the number of slots in the buffer.

44.3.5 Fast Arbitration Module Functional Description - 1st Degree

The fast arbitration module is responsible for all the accesses that go to ESDCTLv2 memory controller, LPDDR/DDR2 SDRAM memory devices. As was described in the sections before, the addr decoding is done at the AXI port gasket, any access that is pointed to the ESDCTL memory controller would be sent to the fast arbitration module. The fast arbitration arbitrates between all masters request, read and write. The overall number of request is 16, 8 request for read and 8 for write. Each request can be valid on the arbitration logic at any time, it is guaranteed by EMIv2 design, that whenever a write request is valid on the arbitration logic all the data is already stored in the write buffer. It means that data is ready on the bus

between the buffer and the specific memory controller whenever a specific request was chosen by the arbitration module. The fast arbitration uses a dynamic priority engine to perform a high performance low latency data path arbitration. The dynamic arbitration uses the following parameters in its engine and calculate the priority of each request accordingly.

44.3.5.1 Page Hit / Miss

LPDDR SDRAM memory device is divided into several banks depending on the specific memory configuration detailed by the memory manufacturer. Before any READ or WRITE commands can be issued to a bank in the LPDDR SDRAM, a row in that bank must be opened. This is accomplished by the ACTIVE command, by which both the Bank and the Row are selected. More than one bank can be active at any time. Once a row is open, a READ or WRITE command could be issued to that row. A subsequent ACTIVE command to another row in the same bank can only be issued after the previous row has been closed. The minimum time interval between two successive ACTIVE commands on the same bank is defined by t_{RC} timing parameter of the LPDRRDR SDRAM device. A Subsequent ACTIVE command to another bank can be issued while the first bank is being accessed, which results in a reduction of total row-access overhead. The minimum time interval between two successive ACTIVE commands on the different banks is defined by t_{RRD} timing parameter of the LPDRRDR SDRAM device.

Therefore, selecting access to the same Bank and the same Row one after the other would be issued in a minimum time at the controller. The arbitration would give high score to a page hit access (same Bank and same Row) than any other access in order to get the highest performance from the controller. (the score level can be configured by the user)

The decoding of the address to bank, row and cs is done in the gasket per each access that arrives from the master. It is later passed on to the “scoring” mechanism in the fast arbitration, in order to be able to calculate the appropriate score for each request. The order of elements on the address is: {cs,bank,row,column}. There is an option to work in “bank interleaving” or “address interleaving” mode. In this mode the order would be: {cs,row,bank,column}. For further details, please refer to the ESDCTLv2 spec.

NOTE

The page hit/mis feature will be functional at clock ratios of up to 1:2 (master:arbitration). At ratios greater than that, the low-frequency master’s page hit/miss score will not be valid, and the master will be selected not according to its page hit/miss score.

44.3.5.2 Last Access Details

In order to perform high density on the LPDDR SDRAM external bus it is needed to consider last access type when choosing the next request (access) on the arbitration to be served. In addition to Page Hit / Miss parameter in the arbitration, a consecutive READ command (READ after READ) would be issued faster than a WRITE after READ. The same is true for a consecutive WRITE command, (WRITE after WRITE).

The dynamic arbitration logic would prefer a consecutive command access in order to get the highest performance from the controller.

44.3.5.3 Basic Priority Configuration

In continue to FBP register description, each master in EMIv2 will have a basic priority configuration based on the user configuration. This basic priority value will be the base value for each master. The dynamic arbitration as described below start its calculation from the basic priority value per each master.

For Elvis project (i.MX51) the recommended configuration of the basic priorities for register FBPM0 (Section 44.2.3.12, F_Basic Priority Reg #0) should be 0x00000203.

44.3.5.4 Priority Calculation

The Fast arbitration module is designed to arbitrate read and write AXI accesses from a maximum number of 8 masters towards the ESDCTL memory controller.

This arbitration takes into account several parameters in order to decide which access to the controller is the most efficient one, out of all the pending requests as was described before.

The arbitration handles a dynamic priority mechanism to allow for a better decision making between the masters. Each master gets a basic priority, which is configurable via an IP register. A dynamic priority' as well as page/hit miss score, are added to this basic priority. Thus a total score is formed and according to this score the request is evaluated by the "decision mechanism" of the arbitration.

The dynamic priority is calculated per each request. Each master has 2 possible requests (1 read and 1 write) and therefore a total of 16 dynamic priorities is evaluated per arbitration. If a request is pending on the "decision mechanism" and other requests are chosen at that time, then after a defined amount of selections (configurable by register), the dynamic priority of this request will increase by a defined number of "points" (also configurable via a register). If, however, a request is chosen, then at that cycle, the dynamic priority of this request will be decreased by the same number of "points" times the number of data words in the burst. (the number of cycles it was taken to served the access)

There is a maximum limit for the dynamic priority. The limit is the same for all masters, and can be configured via a register.

Whenever a master's priority reaches either its limit, it cannot surpass that limit. If theoretically, there is a situation in which the limit needs to be exceeded, the priority will remain at its limit value.

Figure 44-86 illustrates the dynamic priority mechanism for a limit offset of 50 "points":

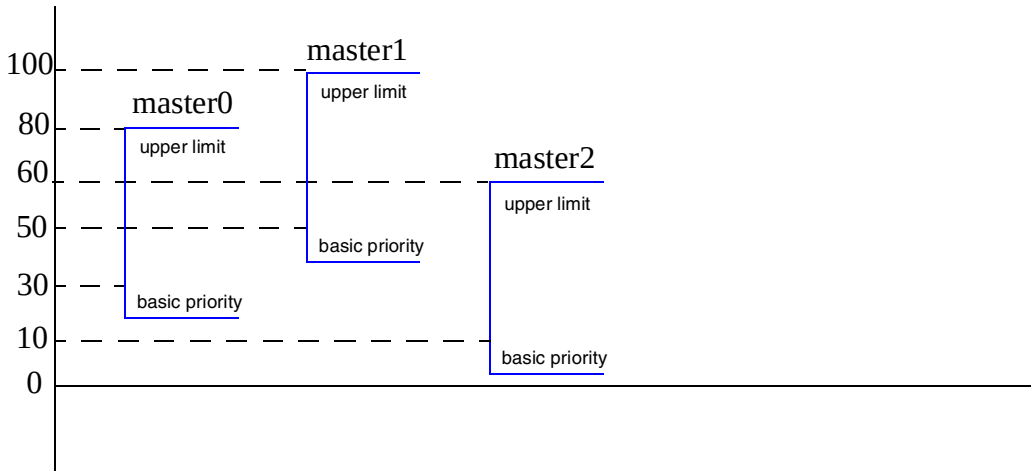


Figure 44-86. Master's Dynamic Priority Scheme

From the above illustration, it is noticeable that one master can "take control" over the arbitration if its basic priority is too high in comparison to the other masters.

There is an option to set a **static priority** to a master. By setting a master's basic priority to "1", the user makes sure that the total score of all of this master's requests will remain fixed on "1". This ensures that the specific master will get access on the bus only when all the other masters are not pending on the "decision mechanism". For that reason, a regular basic priority must have a value of "2" and above.

Therefore, the system configuration needs to be such that the odds of such a situation would be as low as possible. The default settings of the fast arbitration, and the entire M4IF system, were set based on an intensive modeling simulations, changes to these values must be done with extreme care.

In case all the masters have the same score, a simple arbitration logic is used to chose the next request to be served as described in section 1.5.8.

For Elvis project (i.MX51) the recommended configuration of the FPWC register (Section 44.2.3.34, F_Priority Weighting Configuration Register) should be 0x00120125

44.3.5.5 2nd Degree Arbitration (MIF4)

After choosing the best suited request out of the possible 16, it is latched inside a buffer of size 8 (parametric) and there goes through a second degree arbitration. The requests which are stored in this buffer are arbitrated in a similar way to the first degree. Here there's a simpler dynamic priority mechanism. Each request gets a single point for each time another request was chosen over it. When a request is chosen its dynamic counter is reset. The page hit and access hit get smaller amount of points. Not 10 and 5 (default values) like in the first degree, but 4 and 2. (This points are configurable using the MIF4 Scoring register)

The order of accesses from the same master with the same AXI ID is kept. Even if an access has a high score due to page and access hit, it won't be able to bypass an access from the same master with the same AXI ID that arrived before it.

From the second degree arbitration, the selected request is stored in a fifo of depth 4 (parametric). This fifo provides the AXI accesses between the M4IF and the ESDCTLv2.

The 2nd degree arbitration also have 2 more features in its engine:

44.3.5.5.1 Guarding Mechanism

The second degree arbitration is guarded from starvation by using a counter that prevent from a specific request to be stuck inside the 2nd degree buffer for more then a predefined number of clock cycles. For more information about this mechanism please refer to the MIF4 Scoring register.

44.3.5.5.2 Prediction

The second degree arbitration has the ability to predict the {cs,bank,row} that are going to be used by the ESDCTLv2 before the transaction is actually being sent on the AXI bus. By that, we are able to prepare the memory device for future transactions and improve our overall performance. for more information regarding this feature please refer to the ESDCTLv2 spec.

44.3.5.5.3 Recommended configuration

For Elvis project (i.MX51) the recommended configuration of the MIF4 scoring register (Section 44.2.3.14, MIF4 Scoring Register) should be 0x001901A3

44.3.6 Slow Arbitration Module Functional Description

The slow arbitration module is responsible to all access that goes to WEIMv2 memory controller and NFCv2 memory controller, Nor / PSRAM memory devices or Nand devices. As was described in the sections before, the addr decoding is done at the AXI port gasket, any access that is pointed to one of these memory controllers would be sent to the slow arbitration module. The slow arbitration module arbitrates between all masters request, read and write. The overall number of request is 16, 8 request for read and 8 for write. Each request can be valid on the arbitration logic in any time, it is guarantee by EMIv2 design, that whenever a request is valid on the arbitration logic all the data is already stored in the write shared buffer. It means that data is ready on the bus whenever a specific request was chosen by the arbitration module. The slow arbitration uses the "bus divison" mechanism, as described in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)

44.3.7 Internal 1 Memory Arbitration Module Functional Description

The Internal 1 memory arbitration module is responsible to all access that goes to system internal 1 memory controller, like ROM and RAM devices. As was described in the sections before, the addr decoding is done at the AXI port gasket, any access that is pointed to internal 1 memory space would be sent to the internal 1 memory arbitration module. The internal 1 memory arbitration module arbitrates between all masters request, read and write. The overall number of request is 16, 8 request for read and 8 for write. Each request can be valid on the arbitration logic in any time, it is guarantee by EMIv2 design,

that whenever a request is valid on the arbitration logic all the data is already stored in the write shared buffer. It means that data is ready on the bus whenever a specific request was chosen by the arbitration module. The internal 1 memory arbitration uses the “bus divison” mechanism, as described in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)

44.3.8 Internal 2 Memory Arbitration Module Functional Description

The Internal 2 memory arbitration module is responsible to all access that goes to system internal 2 memory controller, like ROM and RAM devices. As was described in the sections before, the addr decoding is done at the AXI port gasket, any access that is pointed to internal 2 memory space would be sent to the internal 2 memory arbitration module. The internal 2 memory arbitration module arbitrates between all masters request, read and write. The overall number of request is 16, 8 request for read and 8 for write. Each request can be valid on the arbitration logic in any time, it is guarantee by EMIv2 design, that whenever a request is valid on the arbitration logic all the data is already stored in the write shared buffer. It means that data is ready on the bus whenever a specific request was chosen by the arbitration module. The internal 2 memory arbitration uses the “bus divison” mechanism, as described in Section 44.3.9, Arbitration Scheme when Masters are in Same Priority (Bus Division)”

44.3.9 Arbitration Scheme when Masters are in Same Priority (Bus Division)

The following section describes the arbitration mechanism when ever the priority of two or more master requests are with the same value. It is most likely that this scheme will be dominant in the slow and internal memory arbitration where the special priority parameters are disables or even when the dynamic priority is disabled.

The low level arbitration scheme exist in all arbitration modules. It is built of several levels of the same logic unit which are connected in a tree built. Figure 44-87 describes the general built of the low level arbitration structure.

Each level of arbitrations (4 levels total) is built of the same logic unit. Each logic unit can be configured to a different arbitration scheme up on 3 input signals that can be partially configured by the user. It is important to mention that the logic unit follows its configured arbitration scheme only when both inputs (masters or requests) are with the same priority level.

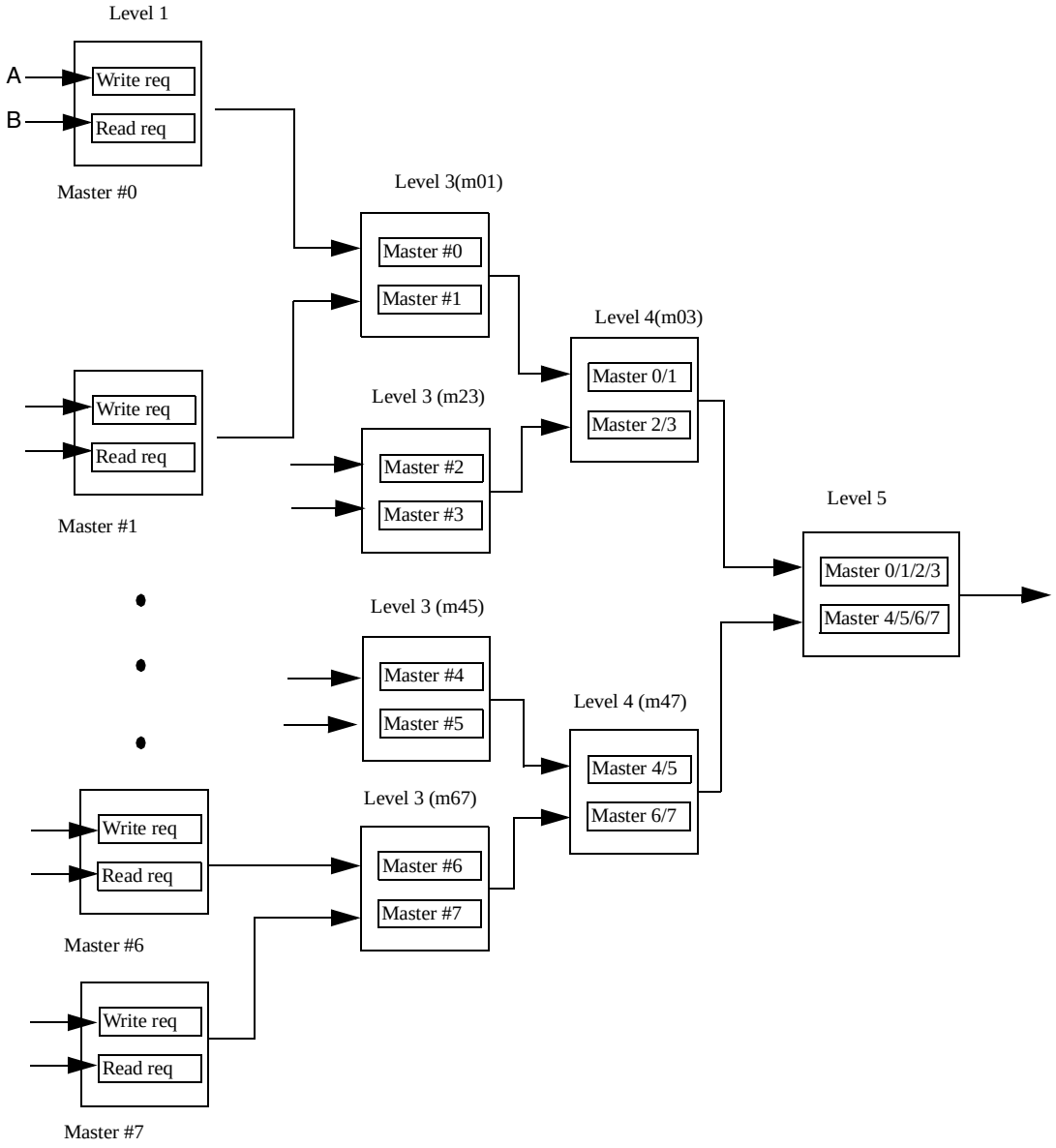


Figure 44-87. Low level arbitration structure

The 3 configuration bits of fast arbitration unit are defined as shown in Table 44-62 below:

Table 44-62. Logic Unit Fast Arbitration Configuration

#	bit 2	bit 1	bit0	arbitration scheme
0	0	0	0	100% priority to B.
1	0	0	1	25% priority to A
2	0	1	0	single 50%
3	0	1	1	double 50%
4	1	0	0	75% priority to A
5	1	0	1	100% priority to A
6	1	1	0	Reserved
7	1	1	1	Reserved

As was mentioned above the user can configure only part of the logic unit levels. Level 1 have a fixed priority scheme configuration and can not be changed while level 3, 4, and 5 can be configured for one of the above arbitration configurations as shown in Table 44-62.

Figure 44-88 describes schematically the arbitration scheme of each condition

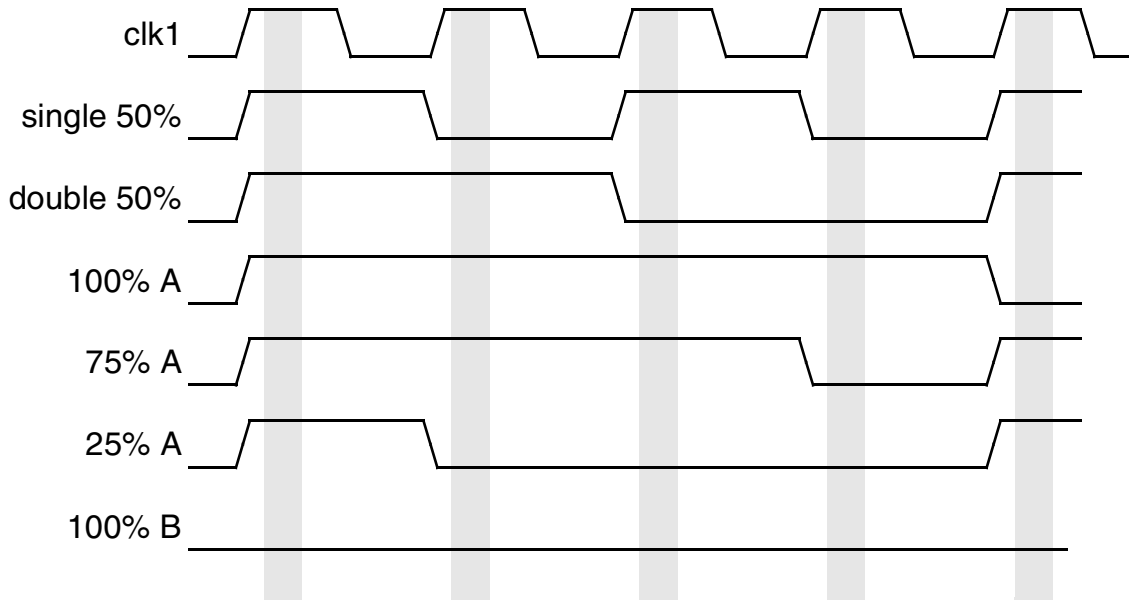


Figure 44-88. Logic units arbitration scheme.

The configuration of the logic units in level 2,3 and 4can be done via the F_Unit_Level_Arbitration_ Register. However, configuration of logic units in level 1 is fixed. Unit level 1 in hard wired configured to ‘double 50%’ state.

For ‘slow’ and ‘intr’ arbitrations, the scheme has changed from previous versions of M4IF. Now there are more configurations that were added, as described in the Table 44-63.Table 44-58

Table 44-63. Slow and Intr Arbitration Bus Division Configuration

#	bit 2	bit 1	bit0	arbitration scheme
0	0	0	0	100% priority to B.
1	0	0	1	12.5% to A.
2	0	1	0	25% to A.
3	0	1	1	37.5% to A.
4	1	0	0	50% - 50%
5	1	0	1	62.5% A.
6	1	1	0	75% to A.
7	1	1	1	100% to A.

44.3.10 WEIM-NFC Downsizer

The slow channel of EMIv2 ultimately splits to 2 memory controllers, WEIM and NFC. These 2 controllers’ AXI data bus width is 32 bits. The slow arbitration has an AXI data bus width of 64. Therefore, a 64bit_to_32bit downsizer is required. Such a downsizer is placed inside the M4IF between the slow arbitration and the AXI interfaces to WEIM and NFC. The downsizer handles bursts of length up to 8 (awlen/areln <= 7) for 64 (awsize/arsize=3) and 32 (awsize/arsize=2) bits. As for 8 and 16 bit accesses, the downsizer only supports single accesses (awlen/arlen=0). The downsizer is also a splitter and can determine the destination of an accesses, whether WEIM or NFC, based on the address of that access.

Note: Single length bursts will always have an output burst type (**awburst_weim** or **awburst_nfc**) INCR (2’b01). The awsize field, in case of 64bit accesses, will change from 64bit (2’b11) to 32bit (2’b10)

Note: Whenever sending burst of size 32 bit through the downsizer, wstrb[7:4] has to be tied to wstrb[3:0]

44.3.10.1 Endianness in Downsizing

The downsizer takes into consideration the endianness of each accesses. If the endianness as LE (0) then the data for addresses with 3 LSbits decoded to: 0,1,2 or 3, will be taken from the lower 32bits out of 64. The data for addresses with 3 LSbits decoded to:4,5,6 or 7 will be taken from the upper 32bits. If the endianness is BE (1) then for addresses with 3 LSbits decoded to: 0,1,2 or 3 data will be taken from the **upper** 32bits, and data corresponding to addresses with 3 LSbits decoded to:4,5,6 or 7 will be taken from the **lower** 32bits.

Note: The information about the destination of an AXI access is not taken into account in the arbitration module. The slow arbitration does not select accesses based on the memory controller’s condition (busy/idle and so on.).

44.3.11 Internal 1 Downsizer

The internal 1 memory channel of EMIv2 ultimately splits to 2 memory controllers, INT1_S0 and INT1_S1. These 2 controllers' AXI data bus width is 32 bits. The internal 1 memory arbitration has an AXI data bus width of 64. Therefore, a 64bit_to_32bit downsizer is required. Such a downsizer is placed inside the M4IF between the internal 1 memory arbitration and the AXI interfaces to INT1_S0 and INT1_S1. The downsizer handles bursts of length up to 8 (awlen/areln <= 7) for 64 (awsize/arsize=3) and 32 (awsize/arsize=2) bits. As for 8 and 16 bit accesses, the downsizer only supports single accesses (awlen/arlen=0). The downsizer is also a splitter and can determine the destination of an accesses, whether INT1_S0 or INT1_S1, based on the address of that access.

Note: Single length bursts will always have an output burst type (**awburst_int1_s0** or **awburst_int1_s1**) INCR (2'b01). The awsize field, in case of 64bit accesses, will change from 64bit (2'b11) to 32bit (2'b10)

Note: Whenever sending burst of size 32 bit through the downsizer, wstrb[7:4] has to be tied to wstrb[3:0]

44.3.11.1 Endianness in Downsizing

The downsizer takes into consideration the endianness of each accesses. If the endianness as LE (0) then the data for addresses with 3 LSbits decoded to: 0,1,2 or 3, will be taken from the lower 32bits out of 64. The data for addresses with 3 LSbits decoded to:4,5,6 or 7 will be taken from the upper 32bits. If the endianness is BE (1) then for addresses with 3 LSbits decoded to: 0,1,2 or 3 data will be taken from the **upper** 32bits, and data corresponding to addresses with 3 LSbits decoded to:4,5,6 or 7 will be taken from the **lower** 32bits.

Note: The information about the destination of an AXI access is not taken into account in the arbitration module. The internal 1 memory arbitration does not select accesses based on the memory controller's condition (busy/idle and so on.)

44.3.12 Clocks

M4IF module contains the following clocks:

- ESDCTLv2 main clock **(up to 200MHz)**
- Slow arbitration clock. **(up to 133MHz)**
- Internal 1 memory arbitration clock. **(up to 166MHz)**
- Internal 2 memory arbitration clock. (**up to 166MHz**)
- 8 x masters clocks, can be asynchronous to EMIv2 arbitration clocks. **(up to 166MHz)**
- Sky Blue interface clock. **(up to 66MHz)**

M4IF assumes all of its 3 main clocks: aclk_fast, aclk_slow, and aclk_intr, are on by default (after reset). If the system wishes to turn one of the clocks off, it must do so by going through the specific lpmd sequence. For example, if the system wishes to turn "aclk_fast" off, it should wake up with this clock on, send an "lpmd_fast" request, wait for "lpack_fast" acknowledge, and only then turn "aclk_fast" off. "lpmd_fast" will have to be asserted for the whole duration of "aclk_fast" being off.

Due to synchronization issues, the general "LPMD"/"DVFS" sequences will only be available when "aclk_fast" is on. **If "aclk_fast" is turned off, these sequences will not work**.

44.3.12.1 Clock Ratios

The “sky blue interface” clock’s (ipg_clk_s) frequency should be shorter (at least 1/2) than that of aclk_fast, aclk_slow, aclk_int1, aclk_int2 and all the master clocks at all times.

44.3.13 Reset

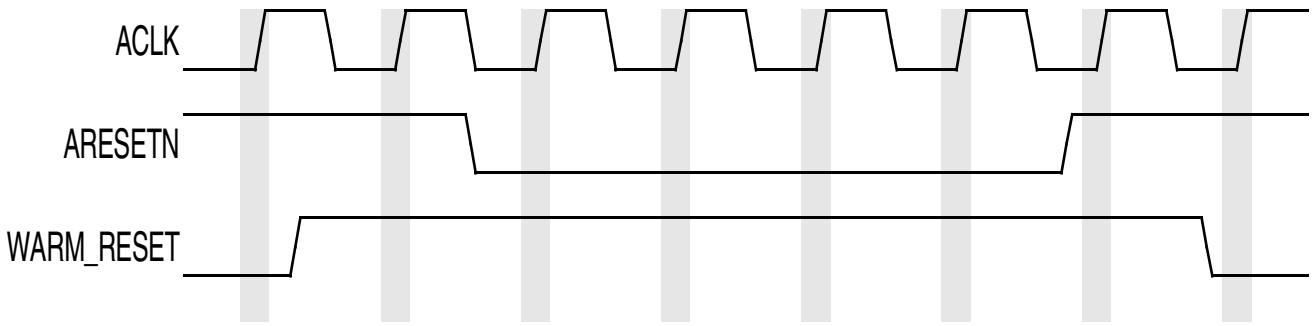
M4IF module as well as the entire EMIv2 module support 3 main reset signals, hard_reset, warm_reset and sw_reset. The reset tree at M4IF module as well as in EMIv2 is built in a way it provides the maximum flexibility to SoC so it can control the reset scheme according to its needs.

44.3.13.1 Software Reset

The software reset is a reset given only to the IP configuration registers. This is done via IP write to bit 31 (SW_RST) in the MCR0 register.

44.3.13.2 Warm Reset

Warm reset is a reset given to the whole system but the configuration register. The warm reset is issued by asserting the regular reset (aresetn) and by asserting the “warm_reset” signal along with it. The “warm_reset” signal must be asserted before the assertion of the “arestn” signal, and must be deasserted after the deassertion of the “aresetn” signal. As illustrated below:



44.3.13.3 EMI Programing Sequence after Warm Reset

This section describes the flow of getting into self-refresh mode before and during warm reset, as well as configuration that is needed to be made to ESDCTL when getting out of warm reset

Please note that after issuing a precharge all via special command register in the exit warm reset, it is recommended by devices to issue a manual auto refresh command, but not necessary.

Figure 44-89. Getting into self refresh before warm_reset assertion

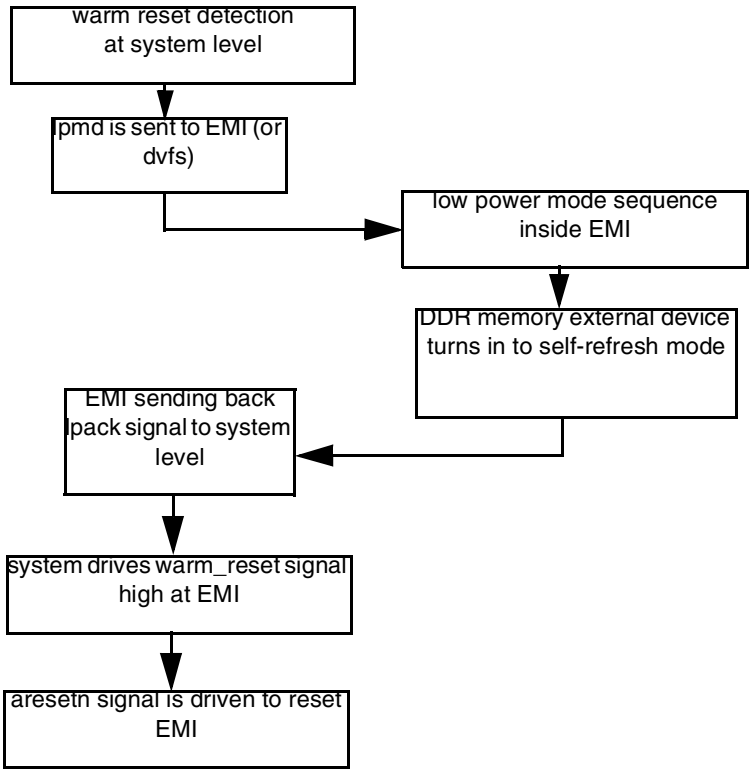
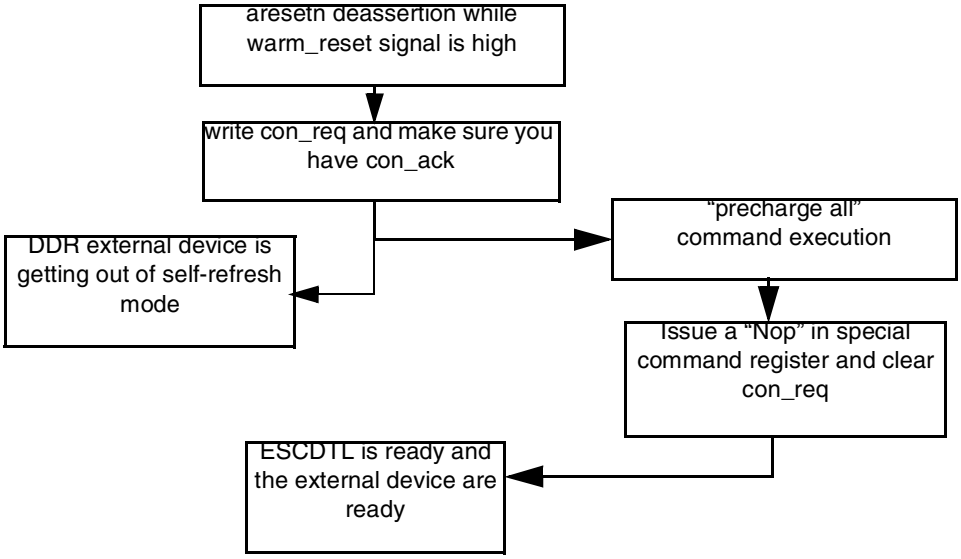


Figure 44-90. Sequence of getting out of self refresh after warm reset



44.3.14 Interrupts

The following interrupts are supported in M4IF module.

- Watermark violation interrupt AP domain.
- Watermark violation interrupt BP domain.
- Master burst length bigger then 8 violation interrupt.

44.3.15 Endianness

M4IF supports both Endianness modes, Little and Big. The memory controllers would take care of the right storage location in the memory, based on Endianness mode and memory data bus size.

The endianness input ports, **endianess_m0** through **endianess_m7**, must not change if the master has outstanding requests inside the M4IF. The master must wait for all read datum and write responses to complete.

44.3.16 AXI Interface Restrictions

This section describes the built-in limitation in current M4IF design that lead to some restriction on the EMIv2 AXI IF support.

> ***Note***: Not complying to these restrictions, will cause an error at best, and an unknown state (possibly dead lock) at worst

44.3.16.1 General Interface Limitations

11. M4IF supports write data interleaved depth of 1 (no interleaving of data is allowed)
12. awcache/arcache - Non-bufferable accesses are supported. In write access there will be a single bit to support bufferable and non-bufferable access types. According to AXI protocol, AWCACHE[0] would reflect the type of the access, AWCACHE[0]=1'b0 means a non-bufferable access and AWCACHE[0]=1'b1 means a bufferable access. The rest of the "cache" signals are not used inside the M4IF. The 'awcache' and 'arcache' output signals (inputs to the memory controller) are set according to the corresponding 'awcache' and 'arcache' inputs from the master.
13. A master that needs to ensure that the written data has reached the destination memory, has to apply non-bufferable access and wait for the response channel.
14. If large block write transfers should be done, it is recommend that the master would drive at least one non-bufferable write access for each ID at the end of the write block transfer to ensure transfer complete at the memory device.
15. In order to keep data access coherency between write and read access of the same master, the response signal will be sent as follow:
    - bufferable write access - BRESP will be sent when last data of the access has entered the memory controller.
    - non-bufferable write access - BRESP will be sent when the data was physically written into the external memory device, and valid response was issued by the controller.
16. The AXI ID bus contains 4 bits total, to reflect the AXI transaction ID as configured by the master. Note: inside M4IF additional 3 bits will be added for internal use only to reflect master number between 0 to 7.

17. Master ID is supported in EMIv2 in a similar way it was used in former versions. Master ID is supported only for address buses and should be considered as ADDR timing. AWMASTER is used for Master ID in write address and ARMASTER for master ID in read address.Each field includes 4 bits max. Note, the master ID is not an AXI requirement and it is supported for AHB masters option mainly.
18. Burst length up to 8 double-words (64 bits), is supported. Access types incr and wrap are supported, fixed access type is not supported. AWSIZE/ARSIZE are 2 bits wide. AWLEN/ARLEN are 3 bits wide.
19. The maximum data bus size is 64 bits.
20. ~~Burst size of 32bits and 64bits are supported, burst size less than 32bits are not supported~~. Burst accesses are supported for all data sizes (8/16/32/64 bits).
21. Single access type is supported in all size range from 8bits up to 64bits.

44.3.16.2 Atomic Accesses

M4IF module support AXI atomic accesses. The atomic access logic is separated to each arbitration module, so that one CS can be locked or under exclusive monitor while other master can continue access another CS on a different arbitration. For example, if one of the master issued a locked access to CSD0 the atomic logic will block any access to the fast channel from any other master (CSD0 & CSD1). At the same time if a master issues other access to WEMv2 CS0 this access would be served as usual.

44.3.16.2.1 AXI Locked Accesses

The M4IF allows a master to perform a lock on a slave region. The lock means that only this master, when using the specific AXI ID of the lock access, can access the locked region.
The master is determined based on EMI master port (0:7) not based on the MASTER_ID field.
All other requests from other masters will be masked till the lock access is done. The lock access request would be active only when the request is served by the arbitration. In case there are other pending accesses of the same master that were issues before the lock access, it would be served first by the arbitration before the lock access take place. The lock is removed upon selecting request from this master, with the lock ID, and ARLOCK/AWLOCK signals indicates an unlocked access (can be either regular or exclusive access).

44.3.16.2.2 Exclusive Accesses

The M4IF deals with exclusive access for each arbitration path separately, each arbitration has its own exclusive access monitor and it is not affected by other exclusive accesses to other arbitrations. Thus, once an exclusive read from a certain master with a specific AXI ID has been selected by the arbitration module, the specific monitor would be active following this access. If another exclusive access occurs to the same arbitration, the monitor would be updated by the new exclusive access and the old access will be dropped from the monitor. In case a write exclusive access related to the first exclusive read access arrives, it would be treated as a regular access and not as an exclusive one. The data of this write exclusive access would not be written and the response will be sent as a regular access.
M4IF supports an exclusive accesses of a single burst type only, that is, the exclusive monitor will monitor a single address only. In case an exclusive access of burst length higher than 1 is done, the exclusive monitor will not be activated.

44.3.16.3 Write Data Interleaving

M4IF has data interleaving depth of 1 as opposed to previous version where the depth was 2. This means that data interleaving is not supported in M4IF.

44.3.17 IPS Interface

All M4IF registers are IPS memory mapped. The registers will be configurable read/write via IPS bus interface only.

The IPS interface unit inside EMIv2 would combined all IPS interfaces from EMIv2 sub modules and M4IF as one of them. The IPS IF unit would route the access to the right sub module based on ips_module_en signal and address decoding.

M4IF registers uses an accessibility policy as described in the EMIv2 spec Chapter.

All M4IF registers are 32bit width. Only IP accesses of 32bit data width are supported.

M4IF IP accesses should be used in one of the following 2 manners:

1. Check all the status bit are idle. Only if all of them idle, an IP access could be made.

Status bit list: FPSS, SPPS, IPSS, I2PSS, M#_PSS (#= Master number)

2. Send a LPMD request to a specific arbitration before configuring it’s registers.

44.3.18 WaterMark Functionality Overview

The WaterMark security feature is used to protect a configurable memory regions (WaterMark space) for up to 8 predefined different CS, 2 CS of ESDCTL and up to 6 of WEIM, no watermark capabilities exist for NFC CS and the internal memory interface. The access (read or write) to the WaterMark space is permitted only if the appropriate control bit of this particular access is high. For read - **m#_wtrmrk_ap(bp)_rd**, for write - **m#_wtrmrk_ap(bp)_wr**. The watermark regions are configurable via the watermark registers where watermark space #0 is for AP masters use and watermark space #1 is for BP masters use. AP masters and BP masters can configure each its own registers only. An AP master cannot configure BP master registers and vice versa. AP and BP masters are being identified by the predefined via at the EMIv2 boundary signals. SoC integration should configure these vias with the right values according to the master ID’s in the system. Only a predefined privilege master (Ap or BP) can configure the watermark registers. Access to the WaterMark space when watermark bits are low is considered as a WaterMark violation. A WaterMark interrupt will be generated in such a violation. In addition the master who cause the violation would not get access, and will receive an error response (DECERR). In case an access was made to watermark space #0 when **wtrmrk_ap** is in low, an AP watermark interrupt will be generated. The same goes for BP watermarked regions. In both cases the address and the master ID of the first master which made a violated access would be kept in the watermark status register till the interrupt is cleared.

Up to two WaterMark spaces can be defined for each predefined CS mapped by the M3IF. The WaterMark space is configurable through a set of configuration registers that contains the WaterMark space start and end address for each chip select. The watermark address has a resolution of 4KByte, means:

- The start address can be set at the end of the first 4KByte at address 0xFFF.
- The start and end address can be set in steps of 4 KByte.

Figure 44-1 illustrates a WaterMark space in SDRAM CSD0 memory region. The red shaded area in the figure is the WaterMark #0 space defined in SDRAM CSD0 memory region while the purple shaded area is the watermark #1 space. These memory regions are accessible only by those masters that their **wtrmrk_ap(bp)** bits are high during the access. Accesses to this region by other masters when wtrmrk_ap(bp) are low generates an interrupt (if enabled through the WaterMark control and status register) and sets the respective WaterMark space (clear by one) status bit. The blue shaded area is the non protected SDRAM CSD0 memory region, means that it is accessible by all masters connected to the M3IF.

NOTE

In case that WEIM operates in collapse mode (CS0 and CS1 are combined as one space) the watermark region definition works as if CS0 and CS1 where separate spaces, start and end address keeps configuration as it was in the non-collapse mode.

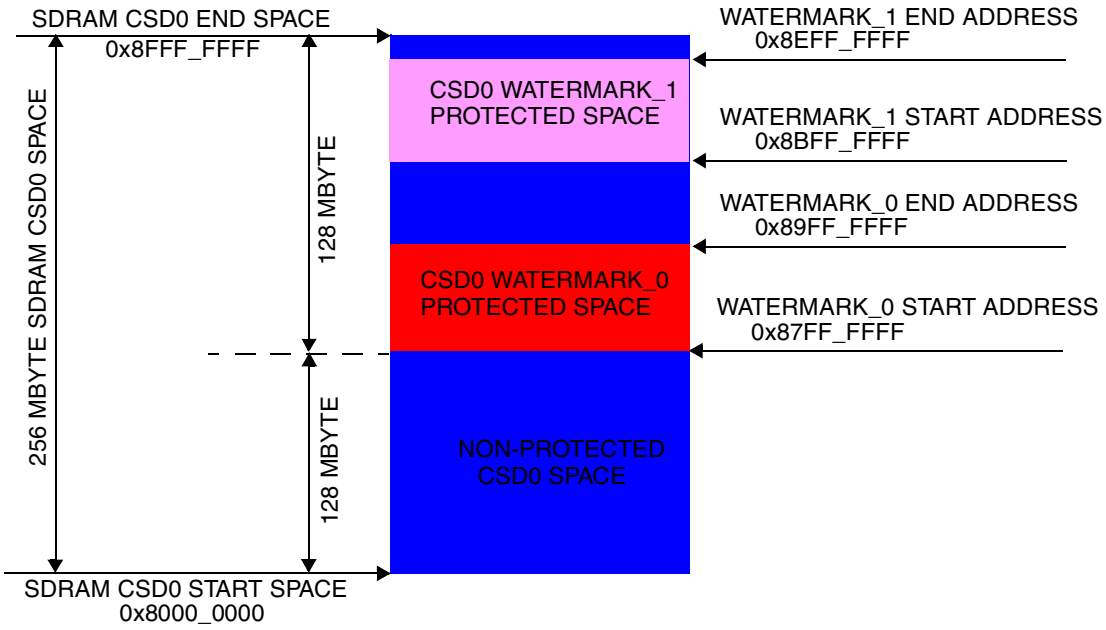


Figure 44-1 SDRAM CS0 WaterMark Space Diagram

44.3.19 Snooping Functionality Overview

The M3IF snooping feature (used by the Image Processor Unit, IPU), monitors and detects write accesses to a configurable window (snooping window) in two of the memory regions mapped by the M3IF. Both memory regions must be in the fast channel (ESDCTLv2). The snooping window base address, memory region, and window size are configurable parameters via the different snooping configuration registers. The snooping window is further divided into 64 equally sized segments.

A detected write access to the snooping window results in:

- The respective segment status bit in SSRL0/SSRL1 and/or SSRH1/SSRH1 registers is set.
- **dma_acceess0** and/or **dma_access1** strobes will assert for a fixed number of fast clock cycles, provided that the snooping segment enable bit is set for the snooped segment. The number of cycles is configurable as well (see register) and is set to 4 by default. The snooping segment enable bit is configured via 2 snooping configuration registers, SERL0/SERL1 and SERH0/SERH1.

It is the software's responsibility to clear the snooped segment status bits (by writing 1 to clear them). The snooped segment status bit will be set for each snooping detection regardless of its value.

44.3.20 Debug Unit

The M4IF has a debug unit in order to allow for better debug capabilities.

The debug unit is consisted of 2 main parts, visibility unit and profiling units.

44.3.20.1 Visibility Unit

This unit muxes and reflects several crucial signals from within M4IF, outside. These signals are described in detail in Section 44.1.3.3.3, Debug Signals". The muxing is done between the 4 channels: fast, slow, internal 1 memory and internal 2 memory. **The user cannot reflect mixed (from different channels) signals using the debug unit as they are from different clock domains**. The desired channel is configured via the MDCR register.

The visibility unit is disabled by default. In order to enable it the user must write to the MDCR register's "DBG_EN" bit. Please refer to Section 44.2.3.32, M4IF Debug Control Register for details.

Note that this unit monitors the signals from within the M4IF, even in a case of fail write exclusive attempt.

44.3.20.2 Profiling Units

There are 4 profiling units inside the debug unit. Each unit is responsible for profiling a single channel (fast, slow, internal memory, and internal 2 memory).

A single profiling unit contains counters which record the number of accesses through the specific channel for each master. The counters are read via IP bus. The information in the IP register does not contain the lower 5 bits of the counter. Meaning, the numbers which are read in the MDSR2-5 registers indicate the number of **groups of 32 accesses**. For example, if master5 performed a total of 274 accesses, the number which will be read in the upper 16 bits of MDSR4 will be: **trunc(274/32)=8**. In order to get the approximate real number of accesses, the user will have to multiply the number in the register by 32. In the above example the outcome will be: **8x32=256**. The error will be **(274-256)/274=7%**. The profiling unit is designed to handle massive use-cases, in which there will be thousands of accesses. In these cases the error will become negligible.

The profiling unit also calculates the maximal dynamic priority of a specific request. The request is selected via the MDCR configuration register. It also calculates the maximal time the master of that request was pending on the bus without getting it. These are read via MDSR0 register.

In order to calculate the average time it took a master to get bus, the profiling unit sums the time (in memory-controller's clock cycles) it took each access to get the bus. This sum can later be read via the MDSR8 register. The average can be calculated by reading the counter for the number of accesses that this master performed (via the MDSR2-5 registers), and dividing the 2 numbers.

Additional 2 registers which contain information about the profiling unit are MDSR6 and MDSR7. MDSR6 contains the total number of accesses a specific request (configured in the MDCR) has accessed the bus. MDSR7 contains the sum of "time for bus" for this specific request. An average of "time for bus" for this request can be calculated by dividing the 2 numbers. There is an option to use these 2 registers not only for a specific request, but for a type of access (read or write). Each master has 2 requests for read and 2 for write. The MDSR6-7 registers can be used to accumulate the statistics for a single request or for both. This is configured by the "SPC_REQ" bit of MDCR.

There is an option to reset all counter of the profiling unit by setting to "1" the "DBG_RST" bit in the MDCR. As long as this bit is asserted all counters will remain 0.

The debug unit's profiling is disabled by default. In order to enable it the user must write to the MDCR register's "DBG_EN" bit. Please refer to Section 44.2.3.32, M4IF Debug Control Register for details.

44.3.21 Buffers Size Table

Table 44-64. Master Buffers

Master #	Master Port Name	Master Port Write Buffers	Master Port Write Command Buffers	Master Port Read Command Buffers
0	IPU	72bx8x4	65bx4	65bx4
1	VPU	72bx8x8	65bx8	65bx16
2	MAX	72bx8x1	65bx1	65bx1
3	ARM	72bx8x6	65bx6	65bx6
4	SDMA	72bx8x1	65bx1	65bx1
5	GPU	72bx8x8	65bx8	65bx12
6	GPU2D	72bx8x4	65bx4	65bx6
7	USB	72bx8x4	65bx4	65bx5

Table 44-65. Slave Buffers

Slave#	Slave Port Name	Slave Port Read Buffers	Slave Pending Address Buffer	Slave Last Address Read buffer	Slave Last address Write buffer
0	FAST	66bx16x8	85bx1	13bx16	13bx3
1	SLOW	66bx4x8	67bx2	13bx4	13bx3
2	INT1	66bx2x8	67bx2	13bx2	13bx3
3	INT2	66bx2x8	67bx3	13bx2	13bx4

44.3.22 Synchronization

There are 12 AXI clock domain involved in the M4IF module. 8 master clocks and 4memory-controller clocks. Clock domain crossings are possible only from master clock to arbitration clock and vice versa. There cannot be a clock domain crossing between master clocks.

44.3.22.1 Synchronization Table

The following table shows the clock-domain relations between all 8 domains of the gaskets/masters, and 4 domains of the arbitrations/slaves.

S - always synchronizer on this path

[] - no synchronizer (domains are synchronized)

NC - No hardware connection at all

Table 44-66. Sync Table

Master #	Master Port Name	FAST	SLOW	INT1	INT2
0	IPU	S	S	S	S
1	VPU	S	S	S	S
2	MAX	S	S	[]	S
3	ARM	S	S	S	S
4	SDMA	S	S	[]	S
5	GPU	S	S	S	NC
6	GPU2D	S	S	S	S
7	USB	S	S	[]	S

44.3.23 Supporting 8/16 Bit Bursts

In previous versions of M4IF (For Marley, Elvis-S, Elvis TO2) bursts of data size 8bit or 16bit were not supported. The only allowed transfers for this data size were singles (bursts with a single data).

This has changed in the latest version of M4IF (Elvis TO3).

It now supports bursts of data size 8/16 bit, and of length up to 8 beat. INCR and WRAP type bursts are supported (according to the AXI protocol spec restrictions).

4 status bits allow indication whether this kind of burst (8/16 bit) occurred. Please refer to: 44.2.3.44, “Master Len Interrupt” for further details.

44.3.24 Dead-Lock Prevention in Read Accesses

M4IF allows flexibility to work in a “parallel routes” scheme. Since it has 4 separate arbitration modules, it can allow parallel work of EMIv2 against all 4 slaves. For example, master #0 can send two requests to

two different memory controllers (say DDR and NFC) and both controllers can handle these requests at the same time without interrupting each other.

This parallel scheme provides enhanced latency hiding when several masters work in parallel against several slaves, or even when a single master works versus multiple slaves.

However, there is a rare theoretical case in which this parallel scheme can cause a dead-lock for read requests. This rare case can only occur if there are 2 masters (or more) that work against the 2 slaves (or more) in a manner of interleaving the read accesses between the slaves. Meaning, a master issues read requests back and forth between two slaves. One request to the first slave and the other to the second slave repeating this over and over. A second master does exactly the same (working with the same slaves).

In order to prevent this case from happening with a 100% certainty, a protection mechanism has been added to the M4IF called “Arbit Stop”.

This mechanism limits a master from sending interleaved read requests. If there is an unfinished access in the master’s read address buffer, M4IF will accept read accesses that are directed to the same slave as the access that is already inside the buffer. If an access to a different location is issued, it will not be inserted to the buffer. The access will wait on the bus until all accesses in the buffer are served and only then will be accepted.

The limitation is only on the read path. It does affect the write path.

Each master can be configured independently to have this limitation or not. Please refer to Section 44.2.3.30, M4IF Control Register #0 for specific details.

Chapter 45
NAND Flash Controller (NFC)

45.1 Introduction

The Nand Flash Controller (NFC) is composed of various control logic units, a 4.5-Kbyte internal RAM and an internal ECC engine. The NFC can interface standard NAND Flash memory devices. Figure 45-1 shows the block diagram of the NAND Flash controller.

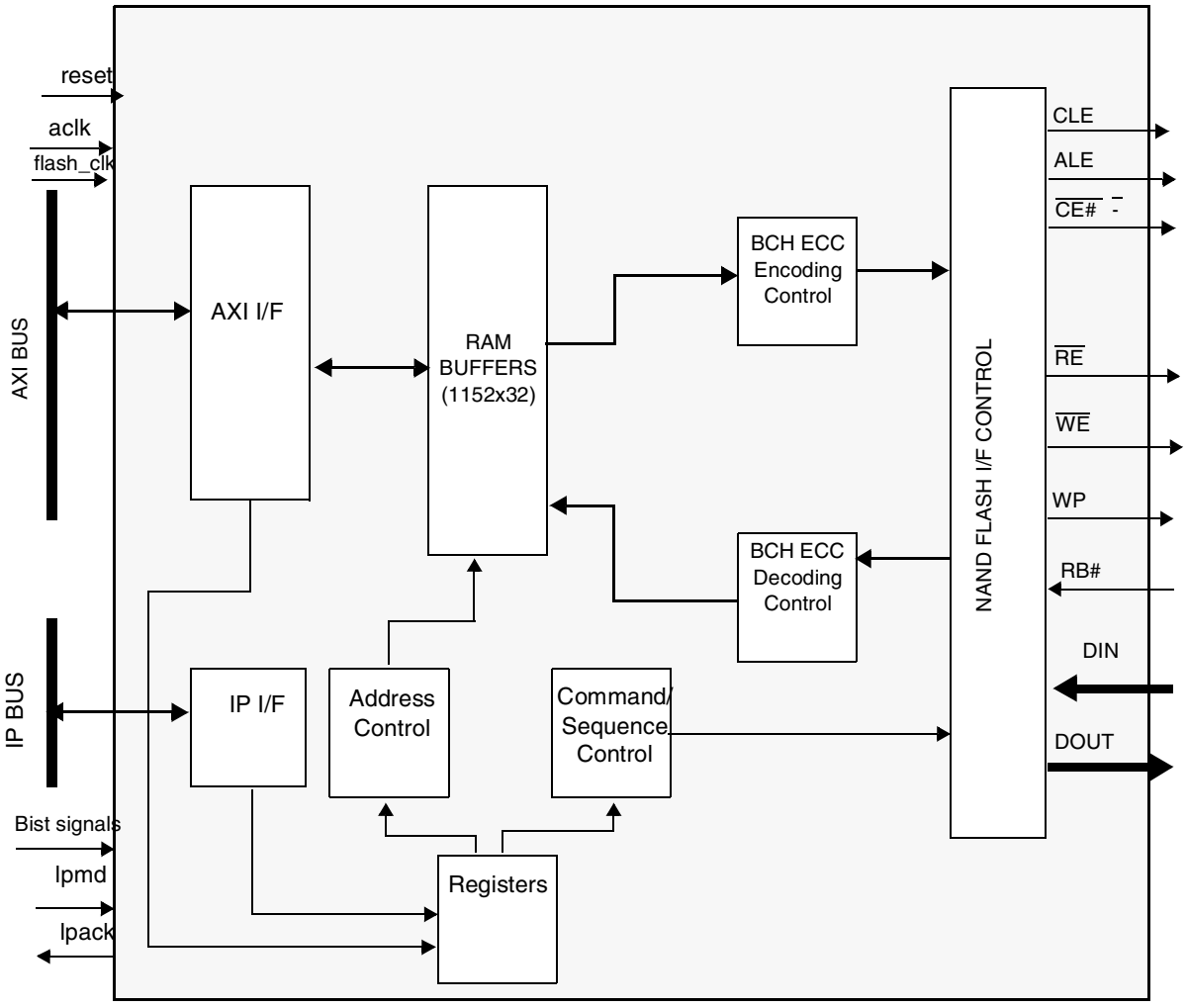


Figure 45-1. NAND Flash Controller Block Diagram

45.2 Overview

The NAND Flash Controller (NFC) can communicate with standard asynchronous NAND Flash devices. It provides a glueless interface to both 8-bits and 16-bits NAND Flash devices with page size of 512B, 2KB, or 4KB.
The NFC is configured by the host through x16/x32 AXI bus and IP bus and according to the configuration it initiates an operation (automatic or atomic) towards the NAND flash device. When the operation is finished the NFC issues an interrupt to the host.
The NFC works in two clock domains. aclk (fast clock) for communication with the AXI and ECC calculations and flash_clk(slow clock) for communication with the NAND flash device.

45.3 Features

The NFC includes the following features:

- x8 / x16 NAND Flash bus.
- Page size of 512B, 2KB and 4KB.
- Internal memory mapped RAM of 4KB + spare of 512B.
- AXI Host interface that can access the internal RAM buffers and several registers. It supports:
  - Read / Write Burst
  - 16-bits / 32-bits bus transfers
- Support atomic (manual) and automatic modes of operation with up to 6 address phases.
- Supports interleaved accesses to up to 4 NAND devices while hiding the busy period of each device
- Supports ECC-BCH(8192) error correction code of 4/8 error bits in 528/538 bytes (512B main data +16B/26B spare).
- Communicates with the system DMA
  - DMA read request after a full page read.
  - DMA write request at the beginning of a full-page programming.
- Multiple Reset
  - AXI reset/Cold Reset/ Warm Reset
- IO pins sharing support
  - Allows sharing of the IO pins with other memory controllers through special arbitration logic.
- Power down handshake mechanism for power saving.

45.4 Restrictions

- In case of working with NAND device of x16bits bus then the ratio between aclk and flash clock must be at least 1:4 (i.e the aclk is at least 4 times faster than the flash_clk).
- In AUTO_COPY_BACK operation with ADD_OP=01 the NFC can copy data only to the following page.
- The address that the NFC drives to the NAND flash device must be aligned to address 0. i.e the columns address must be “0”.
- It is not allowed to configure NUM_OF_ITERATION greater than NUM_OF_DEVICE

45.5 External Signal Description

This section describes the NFC external signals.

45.5.1 Overview

The following signals shown in Table 45-2 are used to configure and control the NFC and its attached Flash device. Signals ending with _b are active-low signals.

Table 45-2. NFC Signal Properties

Name	Function	I/O	Reset
ipp_nfc_ale_out	Flash Address Latch Enable	O	0
ipp_nfc_ce0_out	Flash #0 Chip Enable	O	1
ipp_nfc_ce1_out	Flash #1 Chip Enable	O	1
ipp_nfc_ce2_out	Flash #2 Chip Enable	O	1
ipp_nfc_ce3_out	Flash #3 Chip Enable	O	1
ipp_nfc_cle_out	Flash Command Latch Enable	O	0
ipp_nfc_rb0_in	Flash #0 Ready/Busy	I	1
ipp_nfc_rb1_in	Flash #1 Ready/Busy	I	1
ipp_nfc_rb2_in	Flash #2 Ready/Busy	I	1
ipp_nfc_rb3_in	Flash #3 Ready/Busy	I	1
ipp_nfc_re_out	Flash Read Enable	O	1
ipp_nfc_read_data_in [15:0]	NFC data input from the NAND Flash	I	
ipp_nfc_we_out	Flash Write Enable	O	1
ipp_nfc_wp_out	Flash Write Protect	O	1
ipp_nfc_write_data_out [15:0]	NFC data output towards the NAND Flash	O	0000

45.5.2 Detailed Signal Descriptions

Table 45-3 gives a detailed description of the NFC signals.

Table 45-3. NFC Detailed Signal Descriptions

Signal	I/O	Description
ipp_nfc_ce0_out	O	Flash #0 Chip Enable. This signal indicates the NAND Flash selection. When the NAND Flash device is accessed, this signal is low. This pin also is called CE0#.
ipp_nfc_ce1_out	O	Flash #1 Chip Enable. TThis signal indicates the NAND Flash selection. When the NAND Flash device is accessed, this signal is low. This pin is also called CE1#.
ipp_nfc_ce2_out	O	Flash #2 Chip Enable. This signal indicates the NAND Flash selection. When the NAND Flash device is accessed, this signal is low. This pin is also called CE2#.
ipp_nfc_ce3_out	O	Flash #3 Chip Enable. This signal indicates the NAND Flash selection. When the NAND Flash device is accessed, this signal is low. This pin is also called CE3#.
ipp_nfc_re_out	O	Flash Read Enable. This signal controls the read data from the NAND device. The read data is latched on the rising edge of the signal. This pin is also called RE#.
ipp_nfc_we_out	O	Flash Write Enable.This signal controls the write of commands, address and data to the NAND device. The write is latched on the rising edge of the signal. This pin is also called WE#.
ipp_nfc_cle_out	O	Flash Command Latch Enable. The signal controls the command that is being sent to the command register of NAND device. When active high, commands are latched into the command register of NAND device on the rising edge of the WE# signal. This pin is also called CLE#.
ipp_nfc_ale_out	O	Flash Address Latch Enable. The signal controls address that is being sent to the address register of NAND Flash. When active high, addresses are latched into the NAND device address register of NAND device on the rising edge of the WE# signal. This pin is also called ALE#.
ipp_nfc_wp_out	O	Flash Write Protect. This signal provides inadvertent program/erase protection. This signal is active (held low) during power-up. This pin is is also called WP#.
ipp_nfc_rb0_in	I	Flash #0 Ready/Busy. This signal indicates the status of the NAND Flash operation. When low, it indicates that a program, erase, or read operation of NAND Flash is in process. Upon completion of the process, this signal returns to high state. This pin is also called RB. **Note:** This signal is connected to an open drain output, via a 100 KOhm pull-up resister (outside the external NAND Flash memory device).
ipp_nfc_rb1_in	I	Flash #1 Ready/Busy. This signal indicates the status of the NAND Flash operation. When low, it indicates that a program, erase, or read operation of NAND Flash is in process. Upon completion of the process, this signal returns to high state. This pin is also called RB. **Note:** This signal is connected to an open drain output, via a 100 KOhm pull-up resister (outside the external NAND Flash memory device).

Table 45-3. NFC Detailed Signal Descriptions

Signal	I/O	Description
ipp_nfc_rb2_in	I	Flash #2 Ready/Busy. This signal indicates the status of the NAND Flash operation. When low, it indicates that a program, erase, or read operation of NAND Flash is in process. Upon completion of the process, this signal returns to high state. This pin is also called RB. **Note:** This signal is connected to an open drain output, via a 100 KOhm pull-up resister (outside the external NAND Flash memory device).
ipp_nfc_rb3_in	I	Flash #3 Ready/Busy. This signal indicates the status of the NAND Flash operation. When low, it indicates that a program, erase, or read operation of NAND Flash is in process. Upon completion of the process, this signal returns to high state. This pin is also called RB. **Note:** This signal is connected to an open drain output, via a 100 KOhm pull-up resister (outside the external NAND Flash memory device).
ipp_nfc_write_data_out[15:0]	O	Flash data output. The NFC uses this bus to drive command/address/data to the NAND device.
ipp_nfc_read_data_in[15:0]	I	Flash data input. The NFC uses this bus to latch data from the NAND device during read operation.

45.6 NFC Memory Map and Registers Definition

45.6.1 Memory Map

Table 45-4 shows the whole memory map of the NFC. Table 45-5 shows the NFC-AXI memory map and shows the NFC-IP memory map.

Table 45-4. Memory map

Address	Use	Access
0xAXI_BASE+0x0000–0xAXI_BASE+0x11FF	Internal RAM	R/W
0xAXI_BASE+0x1200–0xAXI_BASE+0x1BFF	AXI Reserved	-
0xAXI_BASE+0x1E00–0xAXI_BASE+0x1E43	AXI Registers	R/W
0xAXI_BASE+0x1E44–0xAXI_BASE+0x1FFF	AXI Reserved	-
0xBASE+0x3000–0xBASE+0x3037	IP Registers	R/W
0xBASE+0x3038–0xBASE+0x3EFC	IP Reserved	-

Table 45-5. NFC AXI Register Memory Map

Address	Register	Reset Value	Reset Value	Section/Page
0xAXI_BASE+0x1E00 (NAND_CMD)	NAND command	R/W	0x0000_0000	45.7.1/45-19
0xAXI_BASE+0x1E04 (NAND_ADD0)	NAND address0	R/W	0x0000_0000	45.7.2/45-20

Table 45-5. NFC AXI Register Memory Map (continued)

Address	Register	Reset Value	Reset Value	Section/Page
0xAXI_BASE+0x1E08 (NAND_ADD1)	NAND address1	R/W	0x0000_0000	45.7.3/45-20
0xAXI_BASE+0x1E0C (NAND_ADD2)	NAND address2	R/W	0x0000_0000	45.7.4/45-21
0xAXI_BASE+0x1E10 (NAND_ADD3)	NAND address3	R/W	0x0000_0000	45.7.5/45-22
0xAXI_BASE+0x1E14 (NAND_ADD4)	NAND address4	R/W	0x0000_0000	45.7.6/45-22
0xAXI_BASE+0x1E18 (NAND_ADD5)	NAND address5	R/W	0x0000_0000	45.7.7/45-23
0xAXI_BASE+0x1E1C (NAND_ADD6)	NAND address6	R/W	0x0000_0000	45.7.8/45-24
0xAXI_BASE+0x1E20 (NAND_ADD7)	NAND address7	R/W	0x0000_0000	45.7.9/45-24
0xAXI_BASE+0x1E24 (NAND_ADD8)	NAND address8	R/W	0x0000_0000	45.7.10/45-25
0xAXI_BASE+0x1E28 (NAND_ADD9)	NAND address9	R/W	0x0000_0000	45.7.11/45-26
0xAXI_BASE+0x1E2C (NAND_ADD10)	NAND address10	R/W	0x0000_0000	45.7.12/45-26
0xAXI_BASE+0x1E30 (NAND_ADD11)	NAND address11	R/W	0x0000_0000	45.7.13/45-27
0xAXI_BASE+0x1E34 (NFC CONFIGURATION1)	NFC configuration	R/W	0x0000_0000	45.7.14/45-27
0xAXI_BASE+0x1E38 (ECC_STATUS_RESULT)	ECC status result	RO	0x0000_0000	45.7.15/45-30
0xAXI_BASE+0x1E3C (STATUS_SUM)	status sum	R/W	0x0000_0000	45.7.16/45-33
0xAXI_BASE+0x1E40 (LAUNCH NFC)	Initiate an NFC operation	R/W	0x0000_0000	45.7.17/45-34

Table 45-6. NFC IP Register Memory Map

Address	Register	Reset Value	Reset Value	Section/Page
0xBASE+0x3000 (NFC_WR_PROTECT)	NAND Flash Write Protection	R/W	0x0000_0000	45.7.18/45-38
0xBASE+0x3000 (NFC_WR_PROTECT)	NAND Flash Write Protection	R/W	0x49249200	45.7.18/45-38

Table 45-6. NFC IP Register Memory Map (continued)

Address	Register	Reset Value	Reset Value	Section/Page
0xBASE+0x3004 (UNLOCK_BLK_ADD0)	Address to Unlock in Write Protection Mode for CS0	R/W	0x0000_0000	45.7.19/45-42
0xBASE+0x3008 (UNLOCK_BLK_ADD1)	Address to Unlock in Write Protection Mode for CS1	R/W	0x0000_0000	45.7.20/45-42
0xBASE+0x300C (UNLOCK_BLK_ADD2)	Address to Unlock in Write Protection Mode for CS2	R/W	0x0000_0000	45.7.21/45-43
0xBASE+0x3010 (UNLOCK_BLK_ADD3)	Address to Unlock in Write Protection Mode for CS3	R/W	0x0000_0000	45.7.22/45-44
0xBASE+0x3014 (UNLOCK_BLK_ADD4)	Address to Unlock in Write Protection Mode for CS4	R/W	0x0000_0000	45.7.19/45-42
0xBASE+0x3018 (UNLOCK_BLK_ADD5)	Address to Unlock in Write Protection Mode for CS5	R/W	0x0000_0000	45.7.20/45-42
0xBASE+0x301C (UNLOCK_BLK_ADD6)	Address to Unlock in Write Protection Mode for CS6	R/W	0x0000_0000	45.7.21/45-43
0xBASE+0x3020 (UNLOCK_BLK_ADD7)	Address to Unlock in Write Protection Mode for CS7	R/W	0x0000_0000	45.7.22/45-44
0xBASE+0x3024 (NFC CONFIGURATION2)	NFC Operation Configuration2	R/W	0x0040_223d	45.7.27/45-47
0xBASE+0x3028 (NFC CONFIGURATION3)	NFC Operation Configuration3	R/W	0x0002_0600	45.7.28/45-49
0xBASE+0x302C (NFC IPC)	NFC IP Control	R/W	0x0000_0000	45.7.29/45-54
0xBASE+0x3030 (AXI_ERR_ADD)	AXI error address	R/W	0x0000_0000	45.7.30/45-56
0xBASE+0x3034 (NFC_DELAY_LINE)	Delay line parameters	R/W	0x0000_0010	45.7.31/45-57

Section 45.7, “Register Descriptions” provides the detailed descriptions for all of the NFC registers.

45.6.2 Internal RAM Address Space and Organization

The internal RAM is divided into 4KB of main area and 1/2KB of spare area. The main area is used to hold the data of the NAND device page in read/program operations and is divided into 8 data buffers (also called data sections) of 512B each. The spare area is used to hold user-reserved data as well as the ECC encoding and is divided into 8 buffers of 64B each. Each main area buffer is associated with spare area buffer. For example spare area buffer 0 is associated with main area buffer 0.

Table 45-7 shows the organization of the main and spare area buffers inside the internal RAM. The host can use all of the spare area except for the ECC code areas. For further information of the internal RAM refer to 45.8.4/45-61

The NFC automatically generates ECC code for both main and spare area during data programming to NAND Flash. When programming/reading a page, the main and spare area buffer number must be selected using the RAM buffer address (RBA) field at NFC_CONFIGURATION1 Spare Area Buffers

Table 45-7. Internal RAM

Address	Use	Access
0xAXI_BASE+0x0000–0xAXI_BASE+0x01FF	Main area Buffer 0	R/W
0xAXI_BASE+0x0200–0xAXI_BASE+0x03FF	Main area Buffer 1	R/W
0xAXI_BASE+0x0400–0xAXI_BASE+0x05FF	Main area Buffer 2	R/W
0xAXI_BASE+0x0600–0xAXI_BASE+0x07FF	Main area Buffer 3	R/W
0xAXI_BASE+0x0800–0xAXI_BASE+0x09FF	Main area Buffer 4	R/W
0xAXI_BASE+0x0A00–0xAXI_BASE+0x0BFF	Main area Buffer 5	R/W
0xAXI_BASE+0x0C00–0xAXI_BASE+0x0DFF	Main area Buffer 6	R/W
0xAXI_BASE+0x0E00–0xAXI_BASE+0x0FFF	Main area Buffer 7	R/W
0xAXI_BASE+0x1000–0xAXI_BASE+0x103F	Spare area Buffer 0 (SB0)	R/W
0xAXI_BASE+0x1040–0xAXI_BASE+0x107F	Spare area Buffer 1 (SB1)	R/W
0xAXI_BASE+0x1080–0xAXI_BASE+0x10BF	Spare area Buffer 2 (SB2)	R/W
0xAXI_BASE+0x10C0–0xAXI_BASE+0x10FF	Spare area Buffer 3 (SB3)	R/W
0xAXI_BASE+0x1100–0xAXI_BASE+0x113F	Spare area Buffer 4 (SB4)	R/W
0xAXI_BASE+0x1140–0xAXI_BASE+0x117F	Spare area Buffer 5 (SB5)	R/W
0xAXI_BASE+0x1180–0xAXI_BASE+0x11BF	Spare area Buffer 6 (SB6)	R/W
0xAXI_BASE+0x11C0–0xAXI_BASE+0x11FF	Spare area Buffer 7 (SB7)	R/W

In case working with a NAND device with spare area of 16B per section of 512B (i.e 512B+spare of 16B, 2KB+ spare of 64B or 4KB + spare of 128B) then only the first 16B of each spare buffer will be used for both user-reserved and ECC code. The number of spare buffers that will be used corresponds the number of main area buffers which are derived from the page size. For example in page size of 4K+ spare of 128B there are 8 main buffers of 512B each and 8 spare buffers of 16B each.

In case working with a NAND device of 4KB + spare of 218B then the size of the first 7 spare buffers (i.e SB0-SB6) will be 26B and the size of the last spare buffer (i.e SB7) will be 36B.

Note:

- In case working with a NAND device with page size of 4K+ spare of 218B with ECC_MODE of 8bits then the last 10B of the last spare buffer (i.e AXI_BASE+0x11DA-AXI_BASE+0x11E3) are not ECC protected. Moreover, in case of 4bits ECC_MODE then the SPAS should be configured to 0x40 (i.e spare of 128B) and the NFC will use only the first 16B (8B for user-reserved and 8B for ECC) for every section of 512B. That means that the NFC will use only 128B of spare out of the 218B available.

Table 45-8 shows the sub-organization of the spare area buffers in case of working with NAND device of 4K+218B. Note that the sub-organization of the first 7 buffers (SB0-SB6) are different from the last buffer (SB7).

Table 45-9 shows the sub-organization of the spare area buffers in case of working with NAND devices of 512B+16B, 2K+64B or 4K+128B (i.e 16B per spare buffer)Register Summary

Table 45-8. Spare Area Buffer for 4K+218B NAND

Address	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0xAXI_BASE+0x1000 (SB0)	Reserved for user(MSB)								Reserved for user(LSB)							
0xAXI_BASE+0x1002 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1004 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1006 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1008 (SB0)	ECC Byte / Reserved for user [1]								ECC Byte / Reserved for user[1]							
0xAXI_BASE+0x100A (SB0)	ECC Byte / Reserved for user[1]								ECC Byte / Reserved for user[1]							
0xAXI_BASE+0x100C (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x100E (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x1010 (SB0)	ECC Byte / Not in Use [2]								ECC Byte / Not in Use[2]							
0xAXI_BASE+0x1012 (SB0)	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
0xAXI_BASE+0x1014 (SB0)	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
0xAXI_BASE+0x1016 (SB0)	ECC Byte / Not in Use r[2]								ECC Byte / Not in Use [2]							
0xAXI_BASE+0x1018 (SB0)	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
0xAXI_BASE+0x101A–0xAXI_BASE_103F (SB0)	Not in Use															

Table 45-8. Spare Area Buffer for 4K+218B NAND (continued)

Address	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0xAXI_BASE+0x1040-0xAXI_BASE+0x107F(SB1)	SB1 - SB6 have same assignment like SB0.															
0xAXI_BASE+0x1080-0xAXI_BASE+0x10BF (SB2)																
0xAXI_BASE+0x10C0-0xAXI_BASE+0x10FF (SB3)																
0xAXI_BASE+0x1100-0xAXI_BASE+0x113F(SB4)																
0xAXI_BASE+0x1140-0xAXI_BASE+0x117F (SB5)																
0xAXI_BASE+0x1180-0xAXI_BASE+0x11BF (SB6)																
0xAXI_BASE+0x11C0-0xAXI_BASE+0x11E3 (SB7)	Reserved for user(MSB)								Reserved for user(LSB)							
	Reserved for user								Reserved for user							
	Reserved for user								Reserved for user							
	Reserved for user								Reserved for user							
	ECC Byte / Reserved for user[1]								ECC Byte / Reserved for user[1]							
	ECC Byte / Reserved for user[1]								ECC Byte / Reserved for user[1]							
	ECC Byte								ECC Byte							
	ECC Byte								ECC Byte							
	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
	ECC Byte / Not in Use [2]								ECC Byte / Not in Use [2]							
	Reserved for user / Not in Use [3]								Reserved for user / Not in Use [3]							
	Reserved for user / Not in Use [3]								Reserved for user / Not in Use [3]							
	Reserved for user / Not in Use [3]								Reserved for user / Not in Use [3]							
	Reserved for user / Not in Use [3]								Reserved for user / Not in Use [3]							
	Reserved for user / Not in Use [3]								Reserved for user / Not in Use [3]							
0xAXI_BASE+0x11E4-0xAXI_BASE+0x11FF	Reserved for user															

[1] When working in 4 bits ECC_MODE, this Byte is an ECC Byte. When working in 8bits ECC_MODE, this Byte is user-reserved

[2] When working in 8 bits ECC_MODE, this Byte is an ECC Byte. When working in 4 bits ECC_MODE, this Byte is not in use.

[3] When working in 8 bits ECC_MODE, this Byte is an unprotected ECC user-reserved. When working in 4 bits ECC_MODE this byte is not used

In case of working with NAND devices of 512B+16B, 2K+64B or 4K+128B (i.e 16B of spare buffer) then only ECC_MODE of 4bits is allowed and then the first 8B of the relevant spare buffers are reserved for user and the rest 8B are reserved for ECC. For example in NAND device of 2K+64B only the first 4 spare buffers are in use.

Table 45-9. Spare Area Buffer for NAND devices of 512B+16B, 2K+64, 4K+128B

Address	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0xAXI_BASE+0x1000 (SB0)	Reserved for user(MSB)								Reserved for user (LSB)							
0xAXI_BASE+0x1002 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1004 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1006 (SB0)	Reserved for user								Reserved for user							
0xAXI_BASE+0x1008 (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x100A (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x100C (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x100E (SB0)	ECC Byte								ECC Byte							
0xAXI_BASE+0x1010– 0xAXI_BASE_103F(SB0)	Not in Use															

Table 45-9. Spare Area Buffer for NAND devices of 512B+16B, 2K+64, 4K+128B

Address	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0xAXI_BASE+0x1040-0xAXI_BASE+0x107F (SB1)	SB1 - SB7 have same assignment like SB0.															
0xAXI_BASE+0x1080-0xAXI_BASE+0x10BF (SB2)																
0xAXI_BASE+0x10C0-0xAXI_BASE+0x10FF (SB3)																
0xAXI_BASE+0x1100-0xAXI_BASE+0x113F (SB4)																
0xAXI_BASE+0x1140-0xAXI_BASE+0x117F (SB5)																
0xAXI_BASE+0x1180-0xAXI_BASE+0x11BF (SB6)																
0xAXI_BASE+0x11C0-0xAXI_BASE+0x11FF (SB7)																

45.6.3 Register Summary

Figure 45-2 shows the key to the register fields and Table 45-10 shows the register figure conventions.

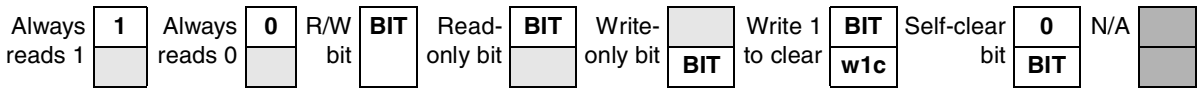


Figure 45-2. Key to Register Fields

Table 45-10. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Reset Values	
0	Resets to zero.
1	Resets to one.

Table 45-11. NFC-AXI Register Summary

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0xAXI_BASE+0x1E00 (NAND_CMD)	R	NAND COMMAND3								NAND COMMAND2							
	W																
	R	NAND COMMAND1								NAND COMMAND0							
	W																
0xAXI_BASE+0x1E04 (NAND_ADD0)	R	NAND ADDRESS0[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E08 (NAND_ADD1)	R	NAND ADDRESS1[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E0C (NAND_ADD2)	R	NAND ADDRESS2[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E10 (NAND_ADD3)	R	NAND ADDRESS3[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E14 (NAND_ADD4)	R	NAND ADDRESS4[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E18 (NAND_ADD5)	R	NAND ADDRESS5[31:0]															
	W																
	R																
	W																

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0xAXI_BASE+0x1E1C (NAND_ADD6)	R	NAND ADDRESS6[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E20 (NAND_ADD7)	R	NAND ADDRESS7[31:0]															
	W																
	R																
	W																
0xAXI_BASE+0x1E24 (NAND_ADD8)	R	NAND ADDRESS1[47:32]															
	W																
	R	NAND ADDRESS0[47:32]															
	W																
0xAXI_BASE+0x1E28 (NAND_ADD9)	R	NAND ADDRESS3[47:32]															
	W																
	R	NAND ADDRESS2[47:32]															
	W																
0xAXI_BASE+0x1E2C (NAND_ADD10)	R	NAND ADDRESS5[47:32]															
	W																
	R	NAND ADDRESS4[47:32]															
	W																

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0xAXI_BASE+0x1E30 (NAND_ADD11)	R	NAND ADDRESS7[47:32]															
	W																
	R	NAND ADDRESS6[47:32]															
	W																
0xAXI_BASE+0x1E34 (NFC CONFIGURATION1)	R	NF_STATUS															
	W																
	R	0	ACTIVE_CS			num_of_iterations				0	RBA			0	NFC_RST	NF_CE	SP_EN
	W																
0xAXI_BASE+0x1E38 (ECC_STATUS_RESULT)	R	NOBER8				NOBER7				NOBER6				NOBER5			
	W																
	R	NOBER4				NOBER3				NOBER2				NOBER1			
	W																
0xAXI_BASE+0x1E3C (STATUS_SUM)	R																
	W																
	R	ECC_SUM								NAND STATUS SUM							
	W																
0xAXI_BASE+0x1E40 (LAUNCH NFC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R				auto_stat	copy_back1	copy_back0	auto_erase	auto_read_cont	auto_read	auto_prog	FDO			FDI	FADD	FCMD
	W																

Table 45-12. NFC-IP Register Summary

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0xBASE+0x3000 (NFC_WR_PROTECT)	R	US7	LS7	LTS 7	US6	LS6	LTS 6	US5	LS5	LTS 5	US4	LS4	LTS4	US3	LS3	LTS 3	US2
	W																
	R	LS2	LTS2	US1	LS1	LTS 1	US0	LS0	LTS 0	BLS		CS2L			WPC		
	W																
0xBASE+0x3004 (UNLOCK_BLK_ADD 0)	R	UEBA0															
	W																
	R	USBA0															
	W																
0xBASE+0x3008 (UNLOCK_BLK_ADD 1)	R	UEBA1															
	W																
	R	USBA1															
	W																
0xBASE+0x300C (UNLOCK_BLK_ADD 2)	R	UEBA2															
	W																
	R	USBA2															
	W																
0xBASE+0x3010 (UNLOCK_BLK_ADD 3)	R	UEBA3															
	W																
	R	USBA3															
	W																
0xBASE+0x3014 (UNLOCK_BLK_ADD 4)	R	UEBA4															
	W																
	R	USBA4															
	W																
0xBASE+0x3018 (UNLOCK_BLK_ADD 5)	R	UEBA5															
	W																
	R	USBA5															
	W																

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name		**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0xBASE+0x301C (UNLOCK_BLK_ADD 6)	R	UEBA6															
	W																
	R	USBA6															
	W																
0xBASE+0x3020 (UNLOCK_BLK_ADD 7)	R	UEBA7															
	W																
	R	USBA7															
	W																
0xBASE+0x3024 (NFC CONFIGURATION2)	R	ST_CMD								SPAS							
	W																
	R	INT_MSK	auto_prog_done_msk	num_addr_phases1		EDC			PPB		ECC_mode	num_addr_phases0	num_cmd_phases	ECC_EN	SYM	PS	
	W																
0xBASE+0x3028 (NFC CONFIGURATION3)	R	0	0	0	0	0	0	0	0	0	0	0	NO_SDMA	FMP			
	W																
	R	rbb_mode	num_of_devices			dma_mode	SBB			0	Bb2R			FW	TOO	ADD_OP	
	W																
0xBASE+0x302C (NFC IPC)	R	INT	auto_prog_done	LPS	RB_B	dma_status		0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CACK	CREQ
	W																
0xBASE+0x3030 (AXI_ERR_ADD)	R																
	W																
	R				AXI ERR ADDRESS												
	W																
0xBASE+0x3034 (NFC_DELAY_LINE)	R																
	W																
	R	NFC_OFF_SEL								NFC_ABS_DEL							
	W																

45.7 Register Descriptions

45.7.1 Nand Flash Command (NAND_CMD)

This register contains the commands to be written during a command phase. The bit assignments for the register are shown in Figure 45-3 and the field descriptions are shown in Table 45-13.

0xAXI_BASE+0x1E00 (NAND_CMD) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND CMD3								NAND CMD2							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND CMD1								NAND CMD0							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-3. NAND Command Register

Table 45-13. NAND Command Field Descriptions

Field	Description
31–24 NAND_CMD3	NAND Flash Command3. second NAND Flash command which will be written to the NAND Flash device during an automatic copy-back operations at the write phase.
23–16 NAND_CMD2	NAND Flash Command2. First NAND Flash command which will be written to the NAND Flash device during an automatic copy-back operations at the write phase.
15–8 NAND_CMD1	NAND Flash Command1. Second NAND Flash command (command confirmation) which will be written to the NAND Flash device during an automatic erase/read/program/copy-back operations. In copy-back operation this command is the second command at the read phase.
7–0 NAND_CMD0	NAND Flash Command0. First NAND Flash command which will be written to the NAND Flash device during an automatic and atomic erase/read/program/copy-back operations. In copy-back operation this command is the second command at the read phase. In atomic(manual) operation only this command will hold the command that will be written to the NAND device.

45.7.2 NAND Flash Address0 (NAND_ADD0)

This register contains the 32 lowest address bits to be written during an address phase to address group0. The bit assignments for the register are shown in Figure 45-4 and the field descriptions are shown in Table 45-14.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD0[0], In other operations the lsb of the column address should be written to NAND_ADD0[0].
  - For example: Assume 4 devices are connected to the NFC, erasing block 0 of device #0 requires writing 48'h0 to NFC_ADDx register. Erasing block 0 requires writing 48'h1 to NFC_ADDx register. Assuming 2KB devices: Programming page 0 requires writing 48'h10000 to NFC_ADDx register (16 lower bits are column address). For furhter examples refer to ADD_OP at 45.7.28/45-49

0xAXI_BASE+0x1E04 (NAND_ADD0) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD0 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD0[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-4. NAND Address0 Register

Table 45-14. NAND Address0 Command Field Descriptions

Field	Description
31–0 NAND_ADD0	NAND Flash address group 0. This defines the lower 32 bits of address group 0. This will be used as the address of the page/block in an automatic and atomic operations.

45.7.3 NAND Flash Address1 (NAND_ADD1)

This register contains the 32 lowest address bits to be written during an address phase to address group1. The bit assignments for the register are shown in Figure 45-5 and the field descriptions are shown in Table 45-15.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD1[0], In other operations the lsb of the column address should be written to NAND_ADD1[0].

0xAXI_BASE+0x1E08 (NAND_ADD1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD1 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD1 [15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-5. NAND Flash Address1 (NAND_ADD1) Register

Table 45-15. NAND Address1 Field Descriptions

Field	Description
31–0 NAND_ADD1	NAND Flash address group 1. This defines the lower 32 bits of address group 1. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.4 NAND Flash Address2 (NAND_ADD2)

This register contains the 32 lowest address bits to be written during an address phase to address group2. The bit assignments for the register are shown in Figure 45-6 and the field descriptions are shown in Table 45-16.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD2[0], In other operations the lsb of the column address should be written to NAND_ADD2[0].

0xAXI_BASE+0x1E0C (NAND_ADD2) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD2 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD2 [15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-6. NAND Flash Address2 (NAND_ADD2) Register

Table 45-16. NAND Address2 Field Descriptions

Field	Description
31–0 NAND_ADD2	NAND Flash address group 2. This defines the lower 32 bits of address group 2. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.5 NAND Flash Address3 (NAND_ADD3)

This register contains the 32 lowest address bits to be written during an address phase to address group3. The bit assignments for the register are shown in Figure 45-7 and the field descriptions are shown in Table 45-17.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD3[0], In other operations the lsb of the column address should be written to NAND_ADD3[0].

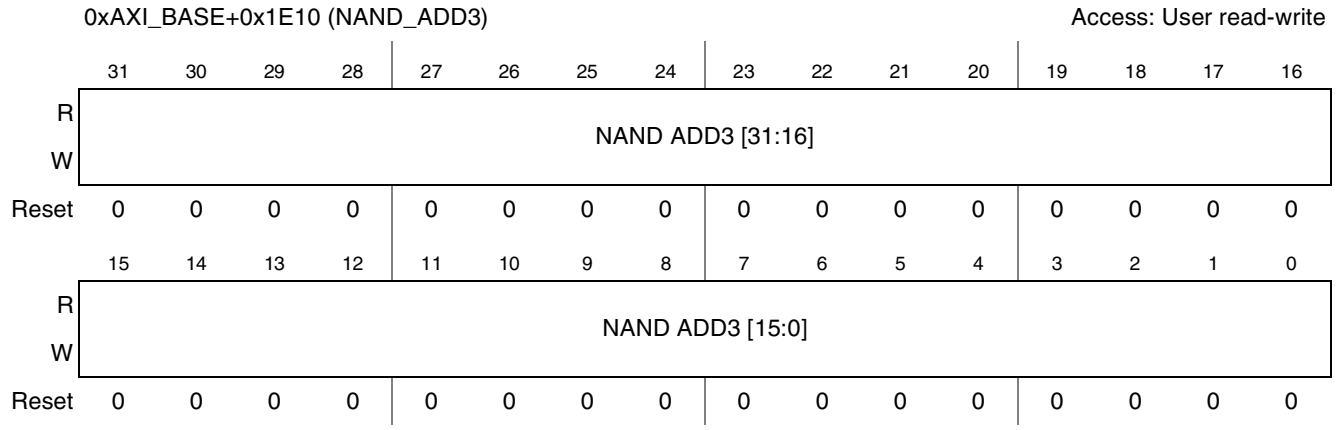


Figure 45-7. NAND Flash Address3 (NAND_ADD3) Register

Table 45-17. NAND Address3 Field Descriptions

Field	Description
31–0 NAND_ADD3	NAND Flash address group 3. This defines the lower 32 bits of address group 3. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.6 NAND Flash Address4 (NAND_ADD4)

This register contains the 32 lowest address bits to be written during an address phase to address group4. The bit assignments for the register are shown in Figure 45-8 and the field descriptions are shown in Table 45-18.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD4[0], In other operations the lsb of the column address should be written to NAND_ADD4[0].

0xAXI_BASE+0x1E14 (NAND_ADD4) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD4 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD4 [15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-8. NAND Flash Address4 (NAND_ADD4) Register

Table 45-18. NAND Address4 Field Descriptions

Field	Description
31–0 NAND_ADD4	NAND Flash address group 4. This defines the lower 32 bits of address group 4. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.7 NAND Flash Address5 (NAND_ADD5)

This register contains the 32 lowest address bits to be written during an address phase to address group5. The bit assignments for the register are shown in Figure 45-9 and the field descriptions are shown in Table 45-19.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD5[0], In other operations the lsb of the column address should be written to NAND_ADD5[0].

0xAXI_BASE+0x1E18 (NAND_ADD5) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD5 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD5 [15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-9. NAND Flash Address5 (NAND_ADD5) Register

Table 45-19. NAND Address Command Field Descriptions

Field	Description
31–0 NAND_ADD5	NAND Flash address group 5. This defines the lower 32 bits of address group 5. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.8 NAND Flash Address6 (NAND_ADD6)

This register contains the 32 lowest address bits to be written during an address phase to address group6. The bit assignments for the register are shown in Figure 45-10 and the field descriptions are shown in Table 45-20.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD6[0], In other operations the lsb of the column address should be written to NAND_ADD6[0].

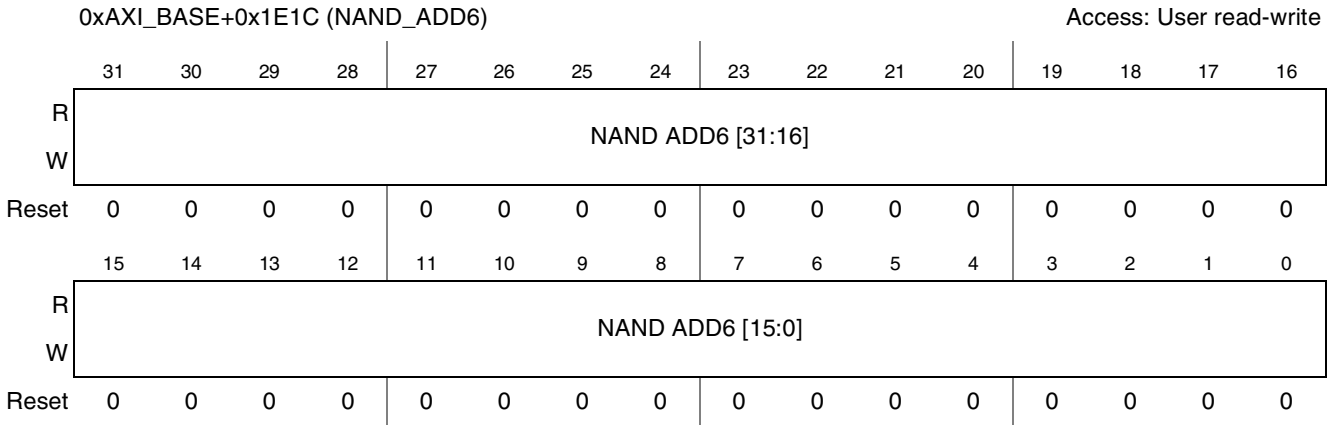


Figure 45-10. NAND Flash Address6 (NAND_ADD6) Register

Table 45-20. NAND Address6 Field Descriptions

Field	Description
31–0 NAND_ADD6	NAND Flash address group 6. This defines the lower 32 bits of address group 6. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.9 NAND Flash Address7 (NAND_ADD7)

This register contains the 32 lowest address bits to be written during an address phase to address group7. The bit assignments for the register are shown in Figure 45-11 and the field descriptions are shown in Table 45-21.

Note:

- The NFC is always aligned to column 0 therefore the address bits of the column suppose to be zero.
- In case of Erase operation the lsb of the page address should be written to NAND_ADD7[0], In other operations the lsb of the column address should be written to NAND_ADD7[0].

0xAXI_BASE+0x1E20 (NAND_ADD7) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD7 [31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD7 [15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-11. NAND Flash Address7 (NAND_ADD7) Register

Table 45-21. NAND Address7 Field Descriptions

Field	Description
31–0 NAND_ADD7	NAND Flash address group 7. This defines the lower 32 bits of address group 7. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.10 NAND Flash Address8 (NAND_ADD8)

This register contains the 16 upper address bits to be written during an address phase to address groups 0,1. The bit assignments for the register are shown in Figure 45-12 and the field descriptions are shown in Table 45-22.

0xAXI_BASE+0x1E24 (NAND_ADD8) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD1 [47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD0[47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-12. NAND Flash Address8 (NAND_ADD8) Register

Table 45-22. NAND Address8 Field Descriptions

Field	Description
31–6 NAND_ADD1[47:32]	NAND Flash address group 1. This defines the upper 16 bits of address group 1. This will be used as the address of the page/block in automatic operations depending on ADD_OP.
15–0 NAND_ADD0[47:32]	NAND Flash address group 0. This defines the upper 16 bits of address group 0. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.11 NAND Flash Address9 (NAND_ADD9)

This register contains the 16 upper address bits to be written during an address phase to address groups 2,3. The bit assignments for the register are shown in Figure 45-13 and the field descriptions are shown in Table 45-23.

0xAXI_BASE+0x1E28 (NAND_ADD9) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD3 [47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD2 [47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-13. NAND Address9 Register

Table 45-23. NAND Address9 Field Descriptions

Field	Description
31–16 NAND_ADD3[47:32]	NAND Flash address group 3. This defines the upper 16 bits of address group 3. This will be used as the address of the page/block in automatic operations depending on ADD_OP.
15–0 NAND_ADD2[47:32]	NAND Flash address group 2. This defines the upper 16 bits of address group 2. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.12 NAND Flash Address10 (NAND_ADD10)

This register contains the 16 upper address bits to be written during an address phase to address groups 4,5. The bit assignments for the register are shown in Figure 45-14 and the field descriptions are shown in Table 45-24.

0xAXI_BASE+0x1E2C (NAND_ADD10) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NAND ADD5 [47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NAND ADD4 [47:32]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-14. NAND Address10 Register

Table 45-24. NAND Address10 Field Descriptions

Field	Description
31–16 NAND_ADD5[47:32]	NAND Flash address group 5. This defines the upper 16 bits of address group 5. This will be used as the address of the page/block in automatic operations depending on ADD_OP.
15–0 NAND_ADD4[47:32]	NAND Flash address group 4. This defines the upper 16 bits of address group 4. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.13 NAND Flash Address11 (NAND_ADD11)

This register contains the 16 upper address bits to be written during an address phase to address groups 6,7. The bit assignments for the register are shown in Figure 45-15 and the field descriptions are shown in Table 45-25.

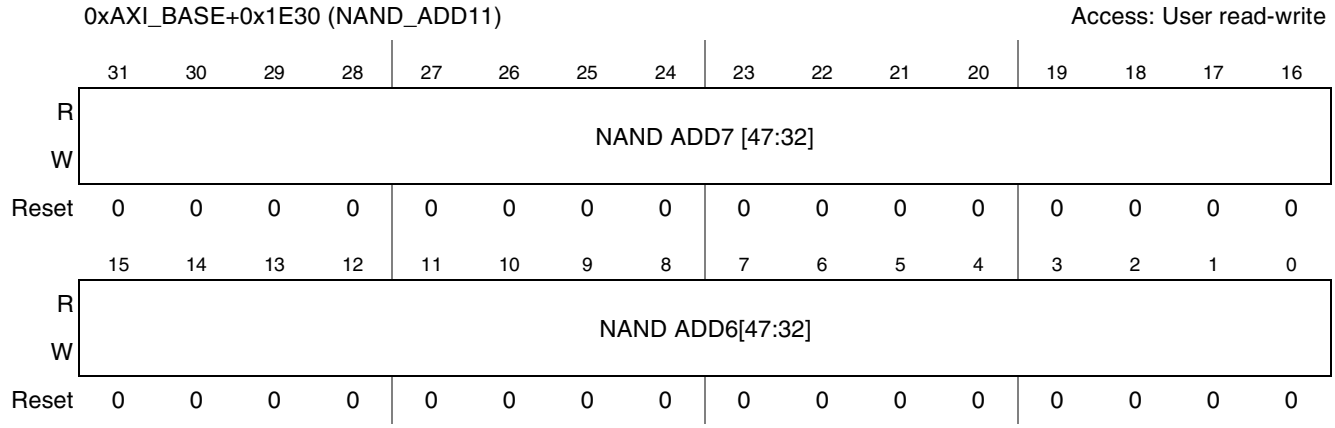


Figure 45-15. NAND Address11 Register

Table 45-25. NAND Address11 Field Descriptions

Field	Description
31–16 NAND_ADD7[47:32]	NAND Flash address group 7. This defines the upper 16 bits of address group 7. This will be used as the address of the page/block in automatic operations depending on ADD_OP.
15–0 NAND_ADD6[47:32]	NAND Flash address group 6. This defines the upper 16 bits of address group 6. This will be used as the address of the page/block in automatic operations depending on ADD_OP.

45.7.14 NFC Configuration (NFC_CONFIGURATION1)

This register specifies interactive configration used by the NFC. The bit assignments for the register are shown in Figure 45-16 and the field descriptions are shown in Table 45-26.

0xAXI_BASE+0x1E34 (NFC CONFIGURATION1) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NF_STATUS															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		ACTIVE_CS			Num_of_iterations					RBA				NFC_RST	NF_CE	SP_EN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-16. NFC_configuration1 Register

Table 45-26. NFC_CONFIGURATION1 Field Descriptions

Field	Description
31-24 Reserved	Reserved
23-16 NF_STATUS	NAND Flash Status. This field holds the last NAND device status that was read by the NFC. This register is updated every time that a read status command is executed (AUTO_STAT, atomic read status).
15 Reserved	Reserved
14-12 ACTIVE_CS	Active CS. This field indicates the NAND device ID to which the NFC will target its operations. This field is being used only during atomic operations (i.e if one of FCMD, FADD, FDI or FDO is set). During an automatic operation the NAND ID is defined based on the ADD_OP register. 000 - access NAND device connected to CS0 001 - access NAND device connected to CS1 010 - access NAND device connected to CS2 011 - access NAND device connected to CS3 100 - access NAND device connected to CS4 101 - access NAND device connected to CS5 110 - access NAND device connected to CS6 111 - access NAND device connected to CS7

Table 45-26. NFC_CONFIGURATION1 Field Descriptions

Field	Description
11–8 NUM_OF_ITERATIONS	Num Of Iterations. Defines how many times NFC will execute AUTO_READ/AUTO_PROG/AUTO_ERASE/AUTO_COPY_BACK without setting an interrupt. When automatic operation bit is set, NFC executes the automatic operation NUM_OF_ITERATIONS+1 times. For example: If NUM_OF_ITERATION is set to 4'h4,and then an AUTO_READ operation is set, then NFC will read 5 pages from the NAND devices. It is the system's responsibility to read the pages content from the NFC's RAM. When NUM_OF_ITERATIONS is greater than 0 and NUM_OF_DEVICES is greater than 0 then an interleaved accesses will be executed and in that case SDMA will be used. For example if NUM_OF_ITERATIONS=1 (i.e 2iterations) and NUM_OF_DEVICES=1(i.e 2 devices) then the NFC will access 1st device then 2nd device and only then issue an interrupt. For furhter examples refer to ADD_OP at 45.7.28/45-49 Note: It is not allowed to configured NUM_OF_ITERATION greater than NUM_OF_DEVICES 4'h0 execute automatic operation 1 time. 4'h1 execute automatic operation 2 times. ... 4'hf execute automatic operation 16 times.
7-6 Reserved	Reserved
6-4 RBA	Ram Buffer Address. Specifies which 1/2K of the internal ram to use when accessing the NAND device. In either Atomic or Automatic operations when NO_SDMA bit is set the value of RBA indicates from which part of the internal RAM to fetch the data in a program operation or to which part of the internal RAM to store the data in a read operation. For example if RBA is set to "100" and the page size is 1/2K (i.e PS is set to "00") and a program operation is configured then the NFC will program the NAND device with the data that is located at the 5th 1/2KB of the internal RAM. In that case it is the user responsiblity to store the requested data at the 5th 1/2KB before starting the program operation. Note that in case the page size is 2KB or 4KB then the NFC will fetch/store the data from/to continous RBA. For example if RBA is set to "000" and page size is 2KB and a program operation is configured then the NFC will program the NAND device with the data that is stored at 1st, 2nd, 3rd, 4th 1/2KB. Note: • In case the page size is 2KB then only RBA of "000" and "100" are allowed and in case the page size is 4KB then only RBA of "000" is allowed. • In case of working with SDMA (i.e NO_SDMA is set to "0") then the NFC will set RBA to "000" in either read or program operations. 000 Use 1st 1/2K of the internal ram. 001 Use 2nd 1/2K of the internal ram 010 Use 3rd 1/2K of the internal ram 011 Use 4th 1/2K of the internal ram 100 Use 5th1/2K of the internal ram 101 Use 6th 1/2K of the internal ram 110 Use 7th 1/2K of the internal ram 111 Use 8th 1/2K of the internal ram
3 Reserved	Reserved

Table 45-26. NFC_CONFIGURATION1 Field Descriptions

Field	Description
2 NFC_RST	NFC Reset. This bit resets the NFC state machines. This bit should be used when issuing a reset command to the NAND Flash device during an operation. When a NAND device received a reset command (0xff) during an operation it aborts its operation and returns to idle. In order to do the same to the NFC, this bit must be set before the reset command is sent. This bit is self cleared. 0 Do not reset the NFC state machine 1 Reset the NFC state machine
1 NF_CE	NAND Flash Force CE. Setting this bit forces the CE# signal that is driven to the NAND Flash device to 0. There are some NAND devices that require that the CE# will be kept low during the busy period. Therefore when working with such a NAND device with RBB_MODE=0 (in that mode the NFC toggles the CE# while perforing a read status during the busy period) NF_CE should set to “1” and then the CE will be forced to “0”. This bit is not self cleared. 0 CE# signal operates normally 1 CE# signal is asserted as long as this bit is set to 1.
0 SP_EN	NAND Flash Spare Enable. This bit determines whether the host will program/read the spare area only or main and spare area together to/from the NAND device. This features is supported only in NAND devices of 1/2KB page size. When this bit is enabled then the NFC will program/read 16B of spare area. **Note:** There is no ECC when SP_EN=1 0 Program/read main and spare area. 1 Program/read spare area only.

45.7.15 ECC Status and Result of Flash Operation (ECC_STATUS_RESULT)

This register shows the number of bits that were detected/corrected for every page section of 528/538 bytes in the main and spare area as a result of the BCH ECC check. The bit assignments for the register are shown in Figure 45-17 and the field descriptions are shown in Table Table 45-27.

0xAXI_BASE+0x1E38 (ECC_STATUS_RESULT) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NOBER8				NOBER7				NOBER6				NOBER5			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NOBER4				NOBER3				NOBER2				NOBER1			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-17. ECC_Status_Result

Table 45-27. ECC_STATUS_RESULT Register Field Description

Field	Description
31-28 NOBER8	Number Of bit Errors for eigth 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved
27-24 NOBER7	Number Of bit Errors for seventh 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved
23-20 NOBER6	Number Of bit Errors for sixth 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved

Table 45-27. ECC_STATUS_RESULT Register Field Description (continued)

Field	Description
19-16 NOBER5	Number Of bit Errors for fifth 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved
15-12 NOBER4	Number Of bit Errors for fourth 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved
11-8 NOBER3	Number Of bit Errors for third528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved

Table 45-27. ECC_STATUS_RESULT Register Field Description (continued)

Field	Description
7-4 NOBER2	Number Of bit Errors for second 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved
3-0 NOBER1	Number Of bit Errors for first 528/538 bytes section. This field shows the number of bit errors in 512 bytes main plus 16/26 bytes spare (totally 528/538 bytes) as a result of the BCH ECC check upon read. 0000 No error 0001>1-bit Error (Correctable Error) 0010> 2-bit Error (Correctable Error) 0011> 3-bit Error (Correctable Error) 0100> 4-bit Error (Correctable Error) 0101> 5-bit Error (Correctable Error for 8bit ecc-mode) 0110> 6-bit Error (Correctable Error for 8bit ecc-mode) 0111> 7-bit Error (Correctable Error for 8bit ecc-mode) 1000> 8-bit Error (Correctable Error for 8bit ecc-mode) 1111 > Uncorrectable error. others > Reserved

45.7.16 Status Summary (STATUS_SUM)

This register shows a summary status of the Nand device as well as a summary result of the ECC check after a page read. The bit assignments for the register are shown in Figure 45-18 and the field descriptions are shown in Table Table 45-28.

0xAXI_BASE+0x1E3C (STATUS_SUM) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ECC_SUM								NAND_STATUS_SUM							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-18. Status_Sum

Table 45-28. STATUS_SUM Register Field Description

Field	Description
31-16 Reserved	Reserved
15-8 ECC_SUM	Ecc Summary report. reports correctable/non-correctable ECC result after a page read for each Nand device. Every Nand device has a bit in this field that states if the last page read from this device had any uncorrectable ECC sections. (bit 0 to Nand0, bit 1 to Nand1, etc). During a page read, NFC checks for error-bits in each section of the page (refer to ECC chapter for more information). If any of the sections had a non-correctable ECC result, then the corresponding bit will be set. If all sectors of the page had correctable ECC errors or no error at all, then the corresponding bit will not be set. If NFC reports an uncorrectable ECC status, the user must clear this field manually.
7-0 NAND_STATUS_SUM	NAND Status Summary. Reports the status of all NAND devices. After NFC executes a status-read operation, as part of the automatic operations (Not status-read of atomic operation (FDO)) to any NAND device, it stores the pass/fail indication of the status register in this field. Each NAND device has 1 bit that stores its status. (bit 0 to Nand0, bit 1 to Nand1, etc). This field is being cleared at any write to LAUNCH_NFC register, thus if an automatic operation was carried more than once to a specific CS, then this field will indicate if any of the operations resulted in an error by the NAND device.

45.7.17 Initiate a Nand Transaction (LAUNCH NFC)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. The bit assignments for the register are shown in Figure 45-19 and the field descriptions are shown in Table 45-29.

	0xAXI_BASE+0x1E40 (LAUNCH NFC)														Access: User read-write	
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	AUTO_START	AUTO_COPY_BACK1	AUTO_COPY_BACK0	AUTO_ERASE	0	AUTO_READ	AUTO_PROG	FDO			FDI	FADD	FCMD
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-19. Launch NFC Register

Table 45-29. Launch NFC register Field Description

Field	Description
31–13	Reserved
12 AUTO_STAT	Automatic Status Read operation. When this bit is set, NFC will automatically perform a status read operation: send status read command that is defined in ST_CMD in NFC_CONFIGURATION2, and then issues 1 pulse of RE_B. The status result will be stored in NF_STATUS field. The summary of the status will be recorded in NAND_STATUS_SUM field. The status-read operation will be executed only once, and it will be not affected from NUM_OF_ITERATION field. 0 Don't execute a status read operation 1 Execute a status read operation.
11 AUTO_COPY_BACK1	Auto copy back sequence to multiple NAND devices. When this bit is set and interleaved mode is configured (i.e NUM_OF_ITERATION and NUM_OF_DEVICES are greater than 0) , NFC will execute a page copy back sequence in interleaved mode as following: <command phase> < address phases> <command phase> <command phase> < address phases> <command phase>. For further information refer to 45.9.1.4/45-65 The commands will be issued from NAND_CMD0, NAND_CMD1, NAND_CMD2 and NAND_CMD3. The source address will be taken from the address group0 and the target address will be taken from address group1. 0 Don't execute a page copy back. 1 Execute a page copy back
10 AUTO_COPY_BACK0	Auto copy back sequence. When this bit is set, NFC will execute a page copy back sequence: <command phase> < address phases> <command phase> <command phase> < address phases> <command phase>. The first command phase will be taken from NAND_CMD0, the second command phase will be taken from NAND_CMD1 the third command phase will be taken from NAND_CMD2 and the last command phase will be taken from NAND_CMD3. In case of ADD_OP other than 01 then the source address will be taken from the address group0 and the target address will be taken from address group1. In case ADD_OP is set to "01" then the NFC will use only address group0 and copy the page to address group0+1 no matter what is address group1. If the user wish to use a device other than device #0 for this operation, then ADD_OP to 2'b01 or 2'b11 must be configured together with the appropriate address. For further information refer to 45.9.1.4/45-65 0 Don't execute a page copy back. 1 Execute a page copy back
9 AUTO_ERASE	Auto erase sequence. When this bit is set, NFC will execute a full block erase sequence: <command phase> < address phases> <command phase> The commands will be issued from NAND_CMD0 and NAND_CMD1. The address will be taken from the active address group. For further information refer to 45.9.1.3/45-65 0 Don't execute a full block erase. 1 Execute a full block erase
8 Reserved	Reserved

Table 45-29. Launch NFC register Field Description

Field	Description
7 AUTO_READ	Auto read sequence. When this bit is set, NFC will execute a full page-read sequence: <command phase> < address phases> <command phase> <busy polling> <data transfer><read status>. The commands will be issued from NAND_CMD0 and NAND_CMD1. The address will be taken from the active address group. If NUM_OF_ITERATIONS is greater than 0, then NFC will continue reading the next page until NUM_OF_ITERATIONS is reached. NFC will monitor the AXI address in order to decide when next page read should start. For further information refer to FMP and to 45.9.1.2/45-65 0 Don't execute a full page read. 1 Execute a full page read
6 AUTO_PROG	Auto program sequence. When this bit is set, NFC will execute a full page-program sequence: <command phase> < address phases> <data transfer> <command phase>. The commands will be issued from NAND_CMD0 and NAND_CMD1. The address will be taken from the active address group. If NUM_OF_ITERATIONS is greater than 0, then NFC will start programming the next page until NUM_OF_ITERATIONS is reached. NFC will monitor the AXI address in order to decide when next page program should start. For further information refer to FPM and to 45.9.1.1/45-64 0 Don't execute a full page program. 1 Execute a full page program
5–3 FDO	NAND atomic Flash Data Output. This bit enables NAND Flash Data Output. 001 One page data out[1]. For further information refer to 45.9.2.5/45-71 010 NAND Flash ID data out. Toggles 6 times the read enable signal (re) and store the returned data at the main buffer according to RBA. For further information refer to 45.9.2.6/45-72 100 NAND Flash Status Register data out. This operation toggle once the read enable signal (re) and store the status in NF_STATUS. For further information refer to 45.9.2.7/45-74 Other options are reserved.
2 FDI	NAND Flash atomic Data Input. This field enables NAND Flash Data Input. For further information refer to 45.9.2.4/45-70 0 No NAND Flash data input operation 1 Enable NAND Flash data input operation
1 FADD	NAND Flash atomic Address Input. This field enables NAND Flash Address Input. For further information refer to 45.9.2.3/45-69 0 No NAND Flash Address input operation 1 Enable NAND Flash Address input operation
0 FCMD	NAND Flash atomic Command Input. This field enables the NAND Flash Command Input. The atomic command will be taken from NAND_CMD0. For further information refer to 45.9.2.2/45-68 0 No NAND Flash Command input operation 1 Allow NAND Flash Command input operation

[1] Page size is determined by SP_EN register bit (main + spare or spare only). It is 528 bytes (main+spare) or 16 bytes (spare) regardless of the nfc_fms setting.

NOTE

When the basic operation is completed, this register is self cleared. Only one of the bit fields can be set at any one given time.

45.7.18 NAND Flash Write Protection (NF_WR_PROT)

This register specifies the Protection command which the controller will perform: Lock, Unlock, or Lock Tight. It also indicates the NAND Flash Write Protection Status. Lock, Unlock or Lock Tight status.

The bit assignments for the register are shown in Figure 45-20 and the field descriptions are shown in Table 45-30.

	0xBASE+0x3000 (NFC_WR_PROTECT)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	US7	LS7	LTS7	US6	LS6	LTS6	US5	LS5	LTS5	US4	LS4	LTS4	US3	LS3	LTS3	US2
W																
RESET	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LS2	LTS2	US1	LS1	LTS1	US0	LS0	LTS0	BLS		CS2L		WPC			
RESET	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0

Figure 45-20. NAND Flash Write Protection Register

Table 45-30. NAND Flash Write Protection Register Field Descriptions

Field	Description
31 US7	Unlocked Status for CS7. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash
30 LS7	Locked Status for CS7. When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
29 LTS7	Lock-tight Status for CS7. Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
28 US6	Unlocked Status for CS6. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash

Table 45-30. NAND Flash Write Protection Register Field Descriptions (continued)

Field	Description
27 LS6	Locked Status for CS6. When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
26 LTS6	Lock-tight Status for CS.: Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
25 US5	Unlocked Status for CS5. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash
24 LS5	Locked Status for CS5. When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
23 LTS5	Lock-tight Status for CS5. Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
22 US4	Unlocked Status for CS4. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash
21 LS4	Locked Status for CS4. When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
20 LTS4	Lock-tight Status for CS4, Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
19 US3	Unlocked Status for CS3. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash

Table 45-30. NAND Flash Write Protection Register Field Descriptions (continued)

Field	Description
18 LS3	Locked Status for CS3.When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
17 LTS3	Lock-tight Status for CS3: Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
16 US2	Unlocked Status for CS2. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash
15 LS2	Locked Status for CS2.When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
14 LTS2	Lock-tight Status for CS2: Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
13 US1	Unlocked Status for CS1. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash
12 LS1	Locked Status for CS1.When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
11 LTS1	Lock-tight Status for CS1: Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
10 US0	Unlocked Status for CS1. This bit indicates whether there are any unlocked blocks in the NAND Flash. This bit gets updated according to WPC. 0 There are no unlocked blocks in NAND Flash 1 there are unlocked block(s) in NAND Flash

Table 45-30. NAND Flash Write Protection Register Field Descriptions (continued)

Field	Description
9 LS0	Locked Status for CS0.When “1” This bit indicate whether all NAND Flash blocks are in locked status or in lock-Unlock status. This bit gets updated according to WPC. 0 not all NAND Flash blocks are in locked status 1 there are unlocked block(s) in NAND Flash
8 LTS0	Lock-tight Status for CS0: Indicates if any of the locked block(s) is (are) have a lock-tight status. This bit gets updated according to WPC. 0 Locked block(s) is (are) not lock-tight 1 Locked block(s) is (are) lock-tight
7–6 BLS	Buffer Lock Set. This field specifies the buffer lock status of first 2KB in the internal buffer (first 2KByte of main data together with 256Bytes of spare data). The other 2KB are always Unlocked. The lock refers only to Nand Accesses. AXI can access all the internal RAM regardless of this field. Any Nand read command, to the first 2KB of the internal RAM, would take place, but, would not be written to the internal RAM (Ecc result on such a read is not valid).(for more details see section 11.10.4/11-46) 00 Locked (default) 01 Locked 10 Unlocked 11 Locked
5–3 CS2L	Chip Select to Lock. When modifying lock status using WPC field, only 1 of the 4 CS supported will be affected. CS2L determines which of the 4 CS will be affected: 000 - CS0 lock status. 001 - CS1 lock status. 010 - CS2 lock status. 011 - CS3 lock status. 100 - CS4 lock status. 101 - CS5 lock status. 110 - CS6 lock status. 111 - CS7 lock status.
2-0 WPC	Write Protection Command. The Command field specifies the operation which the controller will perform. Since, the design might work in a slower clock then IP clock, then, the user must check that his WPC command was updated (by reading relevant LSX/USX/LTSX bits), before any new write to WPC. 100 Unlock NAND Flash block(s) according to given block address range 010 Lock NAND Flash block(s) 001 Lock-tight locked blocks(s) (see also 45.9.6/45-82)

Table 45-31. Write Protect Modes

State	Status bits - US -LS -LTS
Lock - all blocks are locked	010
Unlock-lock - there are unlocked blocks	110

State	Status bits - US -LS -LTS
Unlock-Lockt - there are unlocked blocks: cant change to other state	101
Lockt - all block are locked: cant change to other state	001

45.7.19 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD0)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 0. The bit assignments for the register are shown in Figure 45-21 and the field descriptions are shown in Table 45-32.

	0xBASE+0x3004 (UNLOCK_BLK_ADD0)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA0															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA0															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-21. Unlock_Blk_Add Register

Table 45-32. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA0	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA0	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.20 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD1)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 1. The bit assignments for the register are shown in Figure 45-22 and the field descriptions are shown in Table 45-33.

	0xBASE+0x3008 (UNLOCK_BLK_ADD1)													Access: User read-write		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA1															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-22. Unlock_Blk_Add Register

Table 45-33. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA1	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA1	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.21 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD2)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 2. The bit assignments for the register are shown in Figure 45-23 and the field descriptions are shown in Table 45-34.

	0xBASE+0x300C (UNLOCK_BLK_ADD2)													Access: User read-write		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA2															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA2															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-23. Unlock_Blk_Add Register

Table 45-34. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA2	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA2	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.22 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD3)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 3. The bit assignments for the register are shown in Figure 45-28 and the field descriptions are shown in Table 45-39.

	0xBASE+0x3010 (UNLOCK_BLK_ADD3)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA3															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA3															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-24. Unlock_Blk_Add Register

Table 45-35. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA3	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA3	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.23 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD4)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 3. The bit assignments for the register are shown in Figure 45-28 and the field descriptions are shown in Table 45-39.

	0xBASE+0x3014 (UNLOCK_BLK_ADD4)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA4															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA4															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-25. Unlock_Blk_Add Register

Table 45-36. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA4	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA4	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.24 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD5)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 3. The bit assignments for the register are shown in Figure 45-28 and the field descriptions are shown in Table 45-39.

	0xBASE+0x3018 (UNLOCK_BLK_ADD5)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA5															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA5															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-26. Unlock_Blk_Add Register

Table 45-37. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA5	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA5	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.25 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD6)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 3. The bit assignments for the register are shown in Figure 45-28 and the field descriptions are shown in Table 45-39.

	0xBASE+0x301C (UNLOCK_BLK_ADD6)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA6															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA6															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-27. Unlock_Blk_Add Register

Table 45-38. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA6	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA6	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.26 Address to Unlock in Write Protection Mode (UNLOCK_BLK_ADD7)

Address of block memory in the NAND Flash that is unlocked in Write Protection mode. This register is relevant only when using chip select 3. The bit assignments for the register are shown in Figure 45-28 and the field descriptions are shown in Table 45-39.

	0xBASE+0x3020 (UNLOCK_BLK_ADD7)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	UEBA7															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	USBA7															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-28. Unlock_Blk_Add Register

Table 45-39. Unlock_Blk_Add Register Field Description

Field	Description
31–16 UEBA7	Unlock end Block Address. End address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.
15–0 USBA7	Unlock Start Block Address. Starting address of block memory in the NAND Flash that is unlocked in Write Protection mode. For more details on this, see 45.9.7.4/45-84.

45.7.27 NAND Flash Operation Configuration2 (NFC_CONFIGURATION2)

This register is a configuration register for the NAND Flash device to control the ECC-Enable or Disable, Mask Interrupt, endianess, etc

The bit assignments for the register are shown in Figure 45-29 and the field descriptions are shown in Table 45-40

	0xBASE+0x3024 (NFC CONFIGURATION2)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ST_CMD								SPAS							
RESET	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INT_MSK	auto_prog_done_msk	Num_Adr_phase1		EDC			PPB		ECC_mode	Num_Adr_phases 0	Num_cmd_phases	ECC_EN	SYM	PS	
RESET	0	0	1	0	0	0	0	1	0	0	1	1	1	0	0	1

Figure 45-29. NFC_CONFIGURATION2 Register

Table 45-40. NFC_CONFIGURATION2 Register Field Descriptions

Field	Description
31-24 ST_CMD	Status Command. This is the command NFC will use when executing automatic status-read operations. For most NAND devices this value should be 8'h70
23-16 SPAS	Spare Area Size. This field specifies the size of gthe spare area of the NAND device.This size is in halfwords and refers to a full page. For example when working with a NAND device with page of 2KB + 64B then spare area is 64B so the SPAS should be set to 8'h20.
15 INT_MSK	Mask interrupt Bit. This bit enables the interrupt by masking or not masking the interrupt bit. 0 Mask interrupt is disabled (interrupt enabled) 1 Mask interrupt is enabled (interrupt disabled)
14 AUTO_PROG_DONE_MSK	Auto Program Done Mask. This bit enables the auto_prog_done by masking or not masking the AUTO_PROG_DONE bit. 0 Mask AUTO_PROG_DONE is disabled (auto_prog_done enabled) 1 Mask AUTO_PROG_DONE is enabled (auto_prog_done disabled)
13-12 NUM_ADR_PHASES1	Number Of Address Phases1. he number of address phases need to be executed to the Nand Devices during read/program oeprations. 2'b00 - 3 phases 2'b01 - 4 phases 2'b10 - 5 phases 2'b11 - 6 phases
11–9 EDC	Extra dead cycles. This field specifies number of dead cycles after a read operation, before nfc will allow any other slave to use the shared IO. NFC will delay its ack_en signal according to this field. This will occur any time the NFC stops reading, whether its during a read (because of WEIM request), at the end of a read, or at the end of any read operation. The purpose of this field is to prevent contention on the shared IO when using memories with slow data-to-high-z response (large tRHZ field).
8–7 PPB	Pages Per Block. Indicates how many pages are in 1 Block of the Nand Flash. PPB is used to: - Calculate the next block to erase in AUTO_ERASE. - Calculate in ADD_OP=1/3 where to start the address of the block after extracting the lsb for CS. 00 - 32 pages per block 01 - 64 pages per block 10 - 128 pages per block (default value) 11 - 256 pages per block

Table 45-40. NFC_CONFIGURATION2 Register Field Descriptions (continued)

Field	Description
6 ECC_MODE	ECC_MODE. This bit selects the error correction capabilities, either 4bit correction or 8bit correction. in 4bit ECC_MODE the NFC uses 16Bytes of spare area for for every 512B section of the Nand device. (8B for user specific applications and 8B for ECC). In 8bit ECC_MODE the NFC uses 26B of spare area for every 512B section of the Nand device (12Bytes for user specific application, 14Bytes for ECC). Note that in order to use 8bit ECC_MODE, you must have a NAND device that has at least 26Bytes of spare area for every 512B main section. Note:. -In case working with a NAND device with page size of 4K+ spare of 218B with ECC_MODE of 8bits then the last 10B of the last spare buffer are not ECC protected. Moreover, in case of 4bits ECC_MODE then the SPAS should be configured to 0x40 (i.e spare of 128B) and the NFC will use only the first 16B (8B for user-reserved and 8B for ECC) for every section of 512B. That means that the NFC will use only 128B of spare out of the 218B available. For further information of the ECC algorithm refer to 45.9.4/45-78. 0 - 4bit error correction (default value). 1 - 8bit error correction.
5 NUM_ADR_PHASES0	Number Of Address Phases0. The number of address phases need to be executed to the NAND Device during an AUTO_ERASE operation 0 1 phase less then NUM_ADR_PHASES1 1 2 phases less then NUM_ADR_PHASES1
4 NUM_CMD_PHASES	Number Of Command Phases. he number of command phases need to execute to the NAND Device during AUTO_READ operation(i.e, is read-confirm command is needed or not) 0 1 command phase for reading a page 1 2 command phases for reading a page (read command & read-confirm command)
3 ECC_EN	ECC operation Enable. This bit determines whether ECC operation is executed or bypassed The default is ECC enable. 0 ECC operation is bypassed 1 ECC operation is executed
2 SYM	Symmetric mode. This bit controls the speed of access as well as RE# waveform during read. 0 Two flash clock cycles per access of RE# or WE#, (asymmetric RE waveform). For further information refer to 45.9.5/45-80. 1 One flash clock cycle per access of RE# or WE#, (symmetric RE waveform).
1-0 PS	Page Size. Determines the Nand device page size. 00 - 1/2KB page 01 - 2KB page 10- 4KB page 11 - 4KB page

45.7.28 NAND Flash Operation Configuration3 (NFC_CONFIGURATION3)

This register is a configuration register for the NAND Flash device to control the ECC-Enable or Disable, Mask Interrupt, endianess, etc

The bit assignments for the register are shown in Figure 45-30 and the field descriptions are shown in Table 45-41

	0xBASE+0x3028 (NFC CONFIGURATION3)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												NO_SDMA	FMP			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	rbb_mode	NUM_OF_DEVICES			dma_mode	SBB				Sb2R			FW	TOO	ADD_OP	
W																
RESET	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Figure 45-30. NAND_FLASH_CONFIG Register

Table 45-41. NFC_CONFIGURATION3 Register Field Descriptions

Field	Description
31–21 Reserved	Reserved
20 NO_SDMA	NO_SDMA. When this bit is set, then the system is transferring data to/from the RAM by a different master then the SDMA. Furthermore, it means that dma_wr_req and dma_rd_req signals, are not being used. When SDMA is being used to transfer data to/from the RAM, then it starts each transfer upon assertion of dma_wr_req or dma_rd_req. This activates FMP protection mechanism, and prevents RBA from automatically incrementing. When not using SDMA, then the system must first copy data to the RAM (for program), and only then set the AUTO_PROG bit of the NFC. NFC will automatically increment RBA to point to the next section in the RAM, so if the system wants to program more than 1 page (using NUM_OF_ITERATIONS field) it can do it up to a full use of the RAM. During read, NFC will write the page to the RAM, and again will automatically increment RBA, so if NUM_OF_ITERATIONS is not 4'h0, NFC will automatically start another read sequence and will write it to the next slot in the RAM. NFC will not stop after the RAM was fully used. It is the system responsibility to configure NUM_OF_ITERATIONS field to prevent RAM overflow/underflow. NOTE: Setting this bit disables FMP protection mechanism, so the system must fill the RAM with data before setting AUTO_PROG bit. 0 SDMA is responsible for data transfer to/from NFC RAM. 1 SDMA is not in being used for NFC operations.

Table 45-41. NFC_CONFIGURATION3 Register Field Descriptions (continued)

Field	Description
19-16 FMP	Fifo Mode Protect. During an automatic operation, the system can access NFC's RAM. This means that the system can copy a part of a page to the RAM, then set an automatic program, and only then continue copying the data into the RAM (The system will write into the RAM, and the NFC will read the RAM and send the data to the NAND simultaneously). A similar process can be activated during read, while NFC will start reading the next page before the system finishes to read all the previous page. In order to prevent overwrite of the RAM, the NFC uses a protection mechanism in which the NFC monitors the AXI address, and maintains a safety buffer in which the AXI must be ahead of the NAND operation. This is the system responsibility to make sure that only 1 master is accessing the NFC during such operations otherwise other masters might "confuse" this protection mechanism. This protection mechanism is protecting only the main area of the RAM, so any system operations must access the spare area prior to the main area. (During program, the system should first copy the spare area data into the RAM and only then copy the main data. During read, the system should first read the spare area, and only then read the main data) The buffer size is defined using FMP field: 000 Disable protection buffer. 001 64B 010 128B (default) 011 256B 100 512B 101 1KB (Not to be used with 1/2KB devices) 110 2KB (Not to be used with 1/2KB devices) 111 4KB (To be used only with 4KB devices) **Note:** FMP will be disabled if NO_SDMA bit is set.
15 RBB_MODE	Ready-Busy mode. This bit determines how the NFC monitors the ready-busy signals of the NAND devices only during an automatic operation. During an Atomic operation NFC always monitors the relevant ipp_nfc_rbbx signal. If this bit is cleared, NFC will perform status-read operations to determine the NAND device status. When RBB_MODE is 0 and atomic operation is executed then the user must connect the all ipp_nfc_rbbx signals to "1" (ready). 0 NFC monitors ready-busy status by doing a status-read command (default) 1 NFC monitors ready-busy status by checking ipp_nfc_rbbX signals
14–12 NUM_OF_DEVICES	Num Of Devices. Defines how many NAND devices are connected to the NFC. 000 1 NAND device 001 2 NAND devices 010 3 NAND devices 011 4 NAND devices 100 5 NAND devices 101 6 NAND devices 110 7 NAND devices 111 8 NAND devices

Table 45-41. NFC_CONFIGURATION3 Register Field Descriptions (continued)

Field	Description
11 DMA_MODE	Dma Mode. This bit determines the functionality of the dma_req signals (dma_rd_req and dma_wr_req). If this bit is set, then NFC will output only 1 dma request signal on dma_rd_req which is a logical OR of dma_rd_req & dma_wr_req. The system can determine the request type by reading DMA_STATUS register. If this bit is used, then the system must clear DMA_STATUS field after reading it. 0 NFC outputs 2 dma request signals 1 NFC outputs 1 dma signal which is the logical OR of the 2 dma requests.
10-8 SBB	Status Busy Bit. Indicates which bit in the NAND status register is the Ready/Busy indication. For example: In Samsung's K9K8GU0M bit 6 of the status register is the Ready/Busy indication.
7 Reserved	Reserved
6-4 Sb2R	Status bit to Record. Indicates which bit in the NAND status register is the Pass/Fail indication. For example: In Samsung's K9K8GU0M bit 0 of the status register is the Pass/Fail indication.
3 FW	Flash Width. Determines the Nand Flash IO width (x8/x16). 0 - x16 1 - x8
2 TOO	Two On One. When this bit is set, then NFC assumes there are 2 x8 NAND devices connected to every CS line (One device on the lower byte of ipp_nfc_read_data_in/ipp_nfc_write_data_out, and one device on the upper byte). When using 2 devices on one CS line, then NFC should be configured as if an x16 device is connected to it in order to use all data lines. If 2 devices of 1/2KB page are connected in this manner, then each automatic page-program will transfer 1KB of data from the NFC's RAM to the NAND devices, so RBA field must be 1KB aligned (3'h0, 3'h2, 3'h4 or 3'h6). In atomic operations, NFC transfers only 1/2KB data each time REG_FDI is set, so its the user's responsibility to update RBA field and to set FDI again. If 2 devices of 2KB page are connected in this manner, then each automatic page-program will transfer 4KB of data from the NFC's RAM to the NAND devices, so RBA field must be 4KB aligned (3'h0). In atomic operations, NFC transfers only 2KB data each time FDI is set, so its the user's responsibility to update RBA field and to set FDI again. If 2 devices of 4KB page are connected in this manner, then the user must not use automatic operations since the RAM can not hold 8KB data. Only atomic operations are permitted in this mode. 0 Only 1 device is connected to all CS lines. 1 2 NAND devices are connected to each active CS line.

Table 45-41. NFC_CONFIGURATION3 Register Field Descriptions (continued)

Field	Description
1–0 ADD_OP	Addressing Options. This field defines how NFC handles the addressing of the pages/blocks during an automatic operations. 00 - NFC will use only address group 0 (i.e NAND_ADD0, NAND_ADD8 for all the relevant devices). So in interleaved mode (i.e NUM_OF_ITERATIONS and NUM_OF_DEVICES are greater than 0) then the NFC will act as following: First operation will be sent to CS0 and NFC will drive the address from address group 0 Second operation will be sent to CS1 and NFC will drive the address from address group 0 Third operation will be sent to CS2 and NFC will drive the address from address group 0 and so on untill NUM_OF_ITERATIONS is reached. 01 - NFC will use only address group 0 (i.e NAND_ADD0, NAND_ADD8) and will only use one NAND device as following: The NFC will use the lowest bits of the NAND_ADD0 as the CS for the operations according to NUM_OF_DEVICES . In case of 1 or 2 devices then the lsb (of the page) is used for the CS. In case of 4 devices the 2 lsb (of the page) are used for the CS and in case of 8 devices then the 3 lsb (of the page) are used for CS. Those lowest bits will be removed from the address that will be driven to the NAND device. Moreover for CS calculations, NFC ignores the column address and leaves it as it is. For example: Assume 4 devices are connected to the NFC, erasing block 0 of device #0 requires writing 48'h0 to address group 0 register. Erasing block 0 of device #1 requires writing 48'h1 to NFC_ADD0 register. Assume 2KB page devices: Programming page 0 of device #1 requires writing 48'h10000 to NFC_ADD register (16 lower bits are column address and don't take place in CS-Address extraction) If NUM_OF_ITERATIONS is greater than "0" then the NFC will increment the address group 0 every iteration (For page operations NFC will use the same block address, and will increment the page address. for block operations(erase), NFC will increment the block address) and access the same device (according to the extracted CS) untill NUM_OF_ITERATIONS is reached. For example in Erase: if 4 devices are connected to NFC and PPB is 10 (128 pages per block) and we wish to make two erase iterations (i.e NUM_OF_ITERATIONS=1) of 4th.5th blocks (block number 3,4) of device3 (CS3) then address group 0 supposes to be 48'h603 (2 lsb are used for the CS, 7 bits are used for the page and the rest are used for the block). In the 2nd iteration the address will be incremented to 48'h803 (increment block by 1). For example in Program/Read: if 4 devices are connected to NFC and PPB is 10 (128 pages per block) and we wish to make 3 program/read iterations (i.e NUM_OF_ITERATIONS=2) to/from 2nd,3rd,4th page (page number 1,2,3) of the 2nd block(block number 1) of device3 (CS3) then address group 0 supposes to be 48'h2070000 (16 lsb are used for the column, 2 bits are used for CS, 7 bits are used for the page and the rest are used for the block). In the 2nd iteration the address will be incremented to 48'h20B0000 (increment page by 1) and In the 3rd iteration the address will be incremented to 48'h20F0000 (increment page by 1). Note: - This option can be used only when using 1,2,4 or 8 NAND device. - Only single device will be accessed (No interleaving) even if both NUM_OF_ITERATIONS and NUM_OF_DEVICES are greater than "0" 10 - NFC will use all the address groups. Each device with the associated address group (device0 with address group 0, device1 with address group 1 and so on). So in interleaved mode (i.e NUM_OF_ITERATIONS and NUM_OF_DEVICES are greater than 0) then the NFC will act as following: First operation will be sent to CS0 and NFC will drive the address from address group 0. Second operation will be sent to CS1 and NFC will drive the address from address group 1. Third operation will be sent to CS2 and NFC will drive the address from NAND_ADD3, and so on.

Table 45-41. NFC_CONFIGURATION3 Register Field Descriptions (continued)

Field	Description
	11 - NFC will use all the address groups and also will use the lowest bits of the NAND_ADDx as the CS for the operations. So in interleaved mode (i.e NUM_OF_ITERATIONS and NUM_OF_DEVICES are greater than 0) then the NFC will act as following: First operation will be sent to CSX based on address group 0 (i.e NAND_ADD0, NAND_ADD8). Second operation will be sent to CSY based on address group 1 (i.e NAND_ADD1, NAND_ADD8). Third operation will be sent to CSZ based on address group 2 (i.e NAND_ADD2, NAND_ADD9), and so on. In case of 1 or 2 devices then the lsb (of the page) is used for the CS. In case of 4 devices the 2 lsb (of the page) are used for the CS and in case of 8 devices then the 3 lsb (of the page) are used for CS. Those lowest bits will be removed from the address that will be driven to the NAND device. Moreover for CS calculations, NFC ignores the column address and leaves it as it is. For example in Erase: if 4 devices are connected to NFC and PPB is 10 (128 pages per block) and we wish to make 3 erase iterations (i.e NUM_OF_ITERATIONS=2) the first iteration to the 4th block (block number 3) of device3 (CS3), the second iteration to 2nd block (block number 1) of device1(CS1) and the 3rd to the 1st block (block number 0) of device0 (CS0). Then the address for the 1st iteration will be driven from address group 0 that supposes to be 48'h603 (2 lsb will be used for the CS then 7 bits will be used for the page and the rest are used for the block(which is 3)). The address for the second iteration will be driven from address group 1 that suppposes to be 48'h201 and the address for the 3rd iteration will be driven from address group 2 that supposes to be 48'h0 Note: -This option can be used only when using 1,2,4 or 8 NAND device. In this mode, every address group involved must specify a different CS - It is not allowed to configure NUM_OF_ITERATIONS greater than NUM_OF_DEVICES

45.7.29 NAND Flash IP control (NFC_IPC)

This register holds the IP write handshake registers as well as the interrupt register.

The bit assignments for the register are shown in Figure 45-31 and the field descriptions are shown in Table 45-42

	0xBASE+0x302C (NFC IPC)													Access: User read-write		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	INT	AUTO_PROG_DONE	LPS	RB_B	DMA_STATUS		0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	CACK	CREQ
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-31. NFC_IPC Register

Table 45-42. NFC_IPC Register Field Descriptions

Field	Description
31 INT	Interrupt.The output port ipi_int_nfc is a logic AND between this bit and INT_MSK bit. This bit will rise after NFC finishes an atomic operation (setting any bit in LAUNCH_NFC[5:0]) or in automatic operation (AUTO_READ/AUTO_PROG/AUTO_ERASE/AUTO_COPY_BACK0/1) it will issue the interrupt only after NUM_OF_ITERATIONS is reached. When interrupt is being set after an automatic operation it means that all status registers from all relevant NAND devices have been read and the status is stored in STATUS_SUM register. This bit is not self cleared, so it must be cleared by the user (by writing “0”). 0 NFC didn’t finish its operation. 1 NFC finished its operation and is ready to accept next operation.
30 AUTO_PROG_ DONE	Automatic Program Done. Indicates that SDMA finished transferring all NUM_OF_ITERATIONS pages to the NFC. This allows the software to prepare SDMA with the data of next group of pages to be transferred to the NAND devices. This bit is not self cleared, so it must be cleared by the user (by writing “0”). 0 SDMA didn’t finish transferring NUM_OF_ITERATIONS pages to NFC. 1 SDMA finished transferring num_of_iterations pages to NFC.
29 LPS	Low Power Status. This read-only bit reflects the low power mode of the nfc. 0 - nfc is not in a low power mode 1 - nfc is in a low power mode
28 RB_B	RB_B Status. This read-only bit reflects the RB_B signal of the Nand device. 0 - Nand device is busy. 1 - Nand device is idle.
27–26 DMA_STATUS	Dma Status. This field indicates which dma request signal is asserted. This field is not self cleared, so it must be cleared by the user. 00 All dma request signals are deasserted. 01 dma_wr_req was asserted 10 dma_rd_req was asserted 11 Both dma_rd_req & dma_wr_req were asserted.
25-2 Reserved	Reserved
1 CACK	IP configuration acknowledge. Whenever this bit is set, IP is permitted to configure NFC’s IP registers. 0 - Configuration of NFC’s registers is forbidden. 1 - Configuration of NFC’s registers is permitted.
0 CREQ	IP request to configure NFC. Whenever IP wants to configure NFC it must set this bit to one, and poll on CACK until it is high. Only then configuration is permitted. After configuration is done, the user must deassert this bit. This feature prevents configuring the NFC during an operation and causing an unexpected behaviour. 0 - No request to configure NFC. 1 - A request to configure NFC is vallid.

45.7.30 AXI Error Address (AXI_ERR_ADD)

Address of the first erronous AXI access. The bit assignments for the register are shown in Figure 45-32 and the field descriptions are shown in Table 45-43.

	0xBASE+0x3030 (AXI_ERR_ADD)													Access: User read-write		
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R				AXI_ERR_ADD												
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 45-32. AXI_ERR_ADD Register

Table 45-43. AXI_ERR_ADD Register Field Description

Field	Description
31–13 Reserved	Reserved
12–0 AXI_ERR_ADD	AXI Error Address. Whenever NFC returns an error response on the AXI interface, it will sample the address in which the error occured, and store it in this register for debug purposes. NOTE: Only the first error-address will be stored. If second AXI error response will occur, the second error-address will be lost. Reading from this register, will unlock this register, so, second error-address will be sampled.

45.7.31 Delay-Line (NFC_DELAY_LINE)

The NFC includes 4 delay line units where each unit can adjust the timepoint in which the NFC latches the read data by 1/2 fast clock cycle (aclk) and therefore the 4 delay lines can adjust that timepoint by up to 2 fast clock cycles (aclk)
NFC_DELAY_LINE register represents single delay line unit. By default the NFC latches the data after a fixed 1 flash_clk cycle + 1 fast clock cycle (aclk).
Writing to NFC_DELAY_LINE register will adjust the fast clock cycle delay.
The bit assignments for the register are shown in Figure 45-33 and the field descriptions are shown in Table 45-44.

	0xBASE+0x3034 (NFC_DELAY_LINE)												Access: User read-write			
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
R W	NFC_OFF_DEL								NFC_ABS_DEL							
RESET	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figure 45-33. NFC_DELAY_LINE Register

Table 45-44. NFC_DELAY_LINE Register Field Description

Field	Description
31–16 Reserved	Reserved
15–8 NFC_OFF_DEL	NFC Delay-line Offset delay. The field represents positive and negative numbers using twos compliment representation. This allows positive and negative offsets from the quarter cycle measured value. For example, to add 3 delay units to the measured delay, this field should be 00000011, to subtract 3 delay units, it will be set to 11111101.
7–0 NFC_ABS_DEL	NFC Delay-line Absolute delay. This field decides the specific delay line absolute delay in fractions of a cycle terms. The fraction is process and frequency independent. The delay for the delay line would be (NFC_ABS_DEL / 512) * tCK. So for the default value of 128 we get a quarter cycle delay.

45.8 Functional Description

This section provides the functional description for the NAND Flash Controller.

45.8.1 Reset

There are several options to reset the NFC:
axi_reset - asserting axi_reset_b will reset all axi related functionality, such as axi state machines and axi registers. axi_reset_b resets ip registers as well.
cold reset - asserting of ipp_reset_b will reset all the functionality that is responsible to communicate with the NAND include the ECC engine.

warm_reset - this signal is checked during assertion of axi_reset_b, and if it is high, then all configuration registers will not be reset. This will enable a quick return-to-action reset operation.

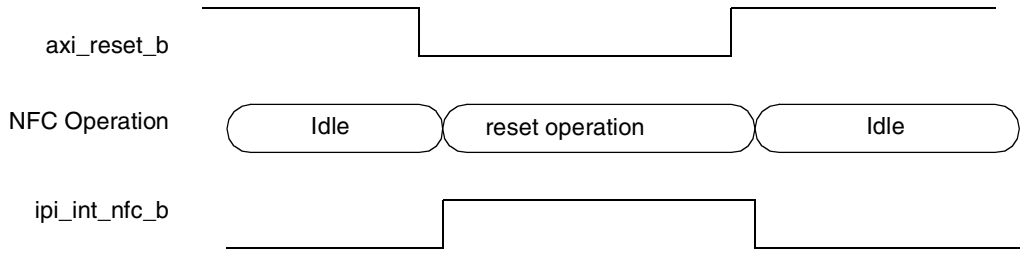


Figure 45-34. Interrupt functionality during reset

45.8.2 NAND Flash I/F Control

The NAND Flash I/F Control generates all the following control signals:

CE# (Flash Chip Enable), RE# (Read Enable for read operations), WE# (Flash Write Enable),
CLE# (Flash Command Latch Enable), ALE (Flash Address Latch Enable).
It monitors the RB_B (Flash Ready/Busy indication) to check if the NAND Flash is in the middle of operation.
Ratio of ipp_flash_clk and aclk should be an integer value and a minimum of 2 (aclk is at least double the frequency of ipp_flash_clk) as long as the clocks are synchronous.
The following will specify the operation of the NFC during program or read:
1/2KB page:
There is a need to configure the address and command sequence before every page of 512B.
Every page is transferred to 1/2KB in the nfc internal RAM buffer so it is important to change RBA register to choose which 1/2KB section of the RAM will be written or read from.
2KB page:
The commands and address cycles occur at the beginning of the program or read sequence. After the command and address sequence, there are four sections of data program/read of 512B each. Every section of 1/2KB is transferred to/from a different 1/2KB in the nfc internal RAM buffer. After command &

address cycles are done, upon kicking off a 2K page (writing 16’h4 or 16’h8 to LAUNCH_NFC register), the NFC will transfer the 2K page from/to its internal RAM to/from the NAND device.
Every page is written to 2KB in the nfc internal RAM buffer so it is important to change RBA register to choose which 2KB section of the RAM will be written or read from. Legal values for RBA field are RBA=0 & RBA=4.
4KB page:
The commands and address cycles occur at the beginning of the program or read sequence. After the command and address sequence, there are eight sections of data program/read of 512B each. Every one of the 1/2KB is transferred to/from different 1/2KB in the nfc internal RAM buffer. After command & address cycles are done, upon kicking off a 4KB (writing 16’h4 or 16’h8 to LAUNCH_NFC register), the NFC will transfer the 4KB page from/to its internal RAM from/to the NAND device.
In 4KB page , it is only legal to write 3’b000 to RBA register.
ECC is always calculated for data section of 1/2KB even if the page size is 2KB/4KB.
Figure 45-35, Figure 45-36, and Figure 45-37 show the NAND Flash read, program, and erase timing diagrams.

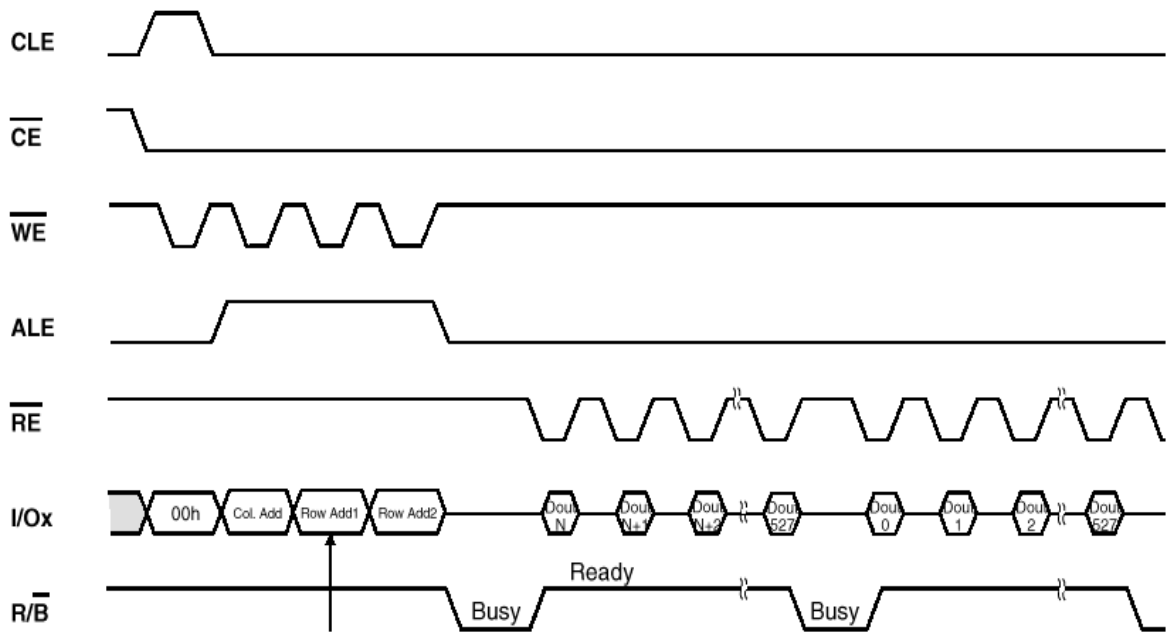


Figure 45-35. Read Operation

1 ALE can be asserted & deasserted separately for each address phase.
2 Number of ALE phases is determined by the user.

PAGE PROGRAM OPERATION

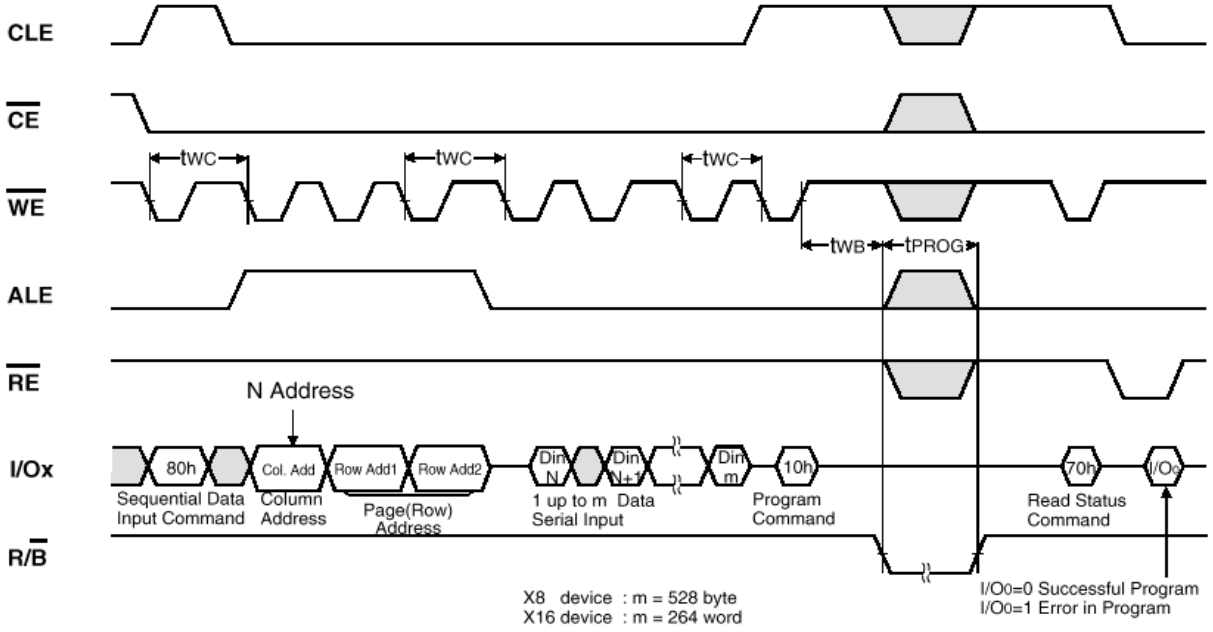


Figure 45-36. Program Operation

BLOCK ERASE OPERATION (ERASE ONE BLOCK)

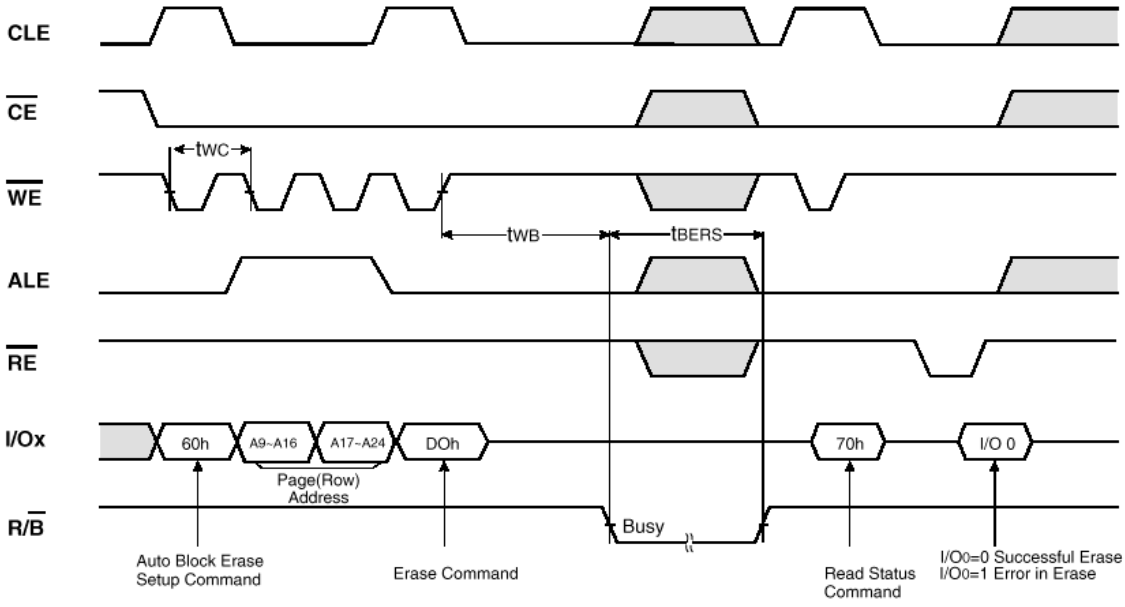


Figure 45-37. Erase Operation

45.8.3 DMA Request Operation

There are two DMA request signals:

- dma_rd_request—triggers after NFC finished reading a page.
- dma_wr_request—triggers either on write to AUTO_PROG bit, or after NFC finished transferring data from NFC's RAM to the NAND device during an automatic program operation, but only if there are more pages to program based on NUM_OF_ITERATIONS field.

The configuration bit DMA_MODE selects between a mode of 2 independent dma-request signals, and a combined signal which asserts both on dma_rd_request triggering & on dma_wr_request triggering. This joined dma-request signal is functional on dma_rd_req pin, whereas dma_wr_req pin will be unused in such a mode.

45.8.4 Internal RAM

The internal RAM Buffer is a 4608 Byte single Port RAM buffer. This memory has 1152 words of 32-bits each, where1024 words are used for the main are buffers (data sections) and the remaining 128 words are allotted to spare area buffers, which are used for ECC (Error Correction) and other user-reserved.
Although the RAM is a single port RAM, the NFC holds a mechanism that allows accessing the RAM from two sources at the same time. For example: When NFC reads data from the RAM in order to program it into the NAND device, the host can write to another location in the RAM and prepare the data for the second page program. This mechanism can be achieved since the Flash side is relatively slow, so, there are idle cycles in which the NFC can service the host. Example of a good use for this feature is that the host instead of writing a full page to the NFC, writes only part of the page, and then launches the program operation. Now the host can continue writing the page to the RAM while NFC is reading it out and sends it to the NAND device. Such a use can almost hide the page transfer time from the host to the NFC.
NOTE: There is a limitation for using the dual-port ability of the RAM. The User must not perform AXI read transactions during NFC-program operation, and the user must not perform AXI write transactions during NFC-read operation (If NFC is reading from the RAM - during program, then the user must not read from the RAM also, and while NFC is writing to the RAM, the user must not write to the RAM either.)
When FMP (Fifo Mode Protect field in NFC_CONFIGURATION3 register) is set, the NFC assumes that the host is acting as the example above. NFC will monitor the RAM and makes sure that there is always enough data in the RAM for the NFC to write. i.e if the host is becoming slow when writing the page into the RAM, then the NFC will pause the program operation and waits until the host continues to write the page. Same monitoring is applied in read operations, when the host reads only a part of a read page, and NFC starts reading another page. The NFC may overwrite the previous page before the host can read it - so the NFC monitors the progress of the host and pauses the read operation before an overwrite can occur. To ensure that this protection mechanism would work properly, the system must make sure that only 1 master is accessing the NFC during the entire operation (for example: SDMA, ARM, etc).
The NFC logically divides the RAM into 8 sections. Each section contains 512B of main data and 64B of spare area. When reading (or programming) from the NAND device, the NFC writes the main data from the NAND device into the main area, and the spare data from the device is written into the spare section. If the NAND device spare data is less than 64B per 512B main data, then the NFC's spare section in the internal RAM will not be fully used. For example in NAND device that has 2KB main data and 64B spare data, then this device has 16B spare data for every 512B main data, then the spare data in the RAM will be located as following: first 16B of spare data in spare buffer 0 ,next 16B in spare buffer 1, and so on.

When using a NAND device in which the spare area size is not fully divided by 512B sections, then the remainder will be located at the last spare section. For example in a NAND device with 4KB main data + 218B spare data, then the division will be 27.25B of spare for each 512B of main data. Then the NFC rounds this number down to the nearest even number, which 26B, and that would be the number of Bytes located in each spare buffer. The remaining Bytes will be added to the last section, meaning that the last spare section will contain 26+10=36B.

45.8.5 AXI Bus Interface

The AXI bus interface is an adapter between ABMA AXI bus and the internal bus.
On the AXI bus side it supports a 16-bit and 32-bit transactions, burst and non burst operations. On the internal bus side it supports 32-bit bus width.
AXI interface supports only INCR accesses (no WRAP nor FIXED), with length of 1-16 data transfers (defined by arlen/awlen of 4 bit).
AXI domain is divided to 2 regions: internal RAM & AXI registers. Any access to AXI registers, must be 32bit wide, with length of 1 data transfer (single access).

45.8.5.1 LPMD/ DVFS

DVFS is supported using a handshake with the system. The system asserts lpmd signal to request a power down. NFC will respond with assertion of lpack, as soon as it becomes idle from its current operation. lpack assertion signals to the HOST that the NFC is ready for power down.
lpack will be asserted if there is no active axi transaction and there is no active flash accesses.
The NFC holds a register in IP domain (LPS) that indicates the status of the lpack signal.

45.8.5.2 Burst Access Support

Not all AXI burst transactions are supported. Table 45-45 lists NFC supported access burst types.

Table 45-45. NAND Flash Burst Access Support

HBURST	BURST TYPE	SUPPORTED	Description
00	FIXED	No	Fixed address burst
01	INCR	Yes	Incrementing-address burst
10	WRAP	No	Incrementing-address burst that wraps at wrap boundary
11	Reserved	Yes	-

Note:

NFC supports bursts of 16/32-bit words only. Bursts of byte words (8-bits) are not supported.

45.8.6 IP interface

The IP interface is an adapter between IP protocol and internal control registers.
NFC's registers are divided between AXI domain & IP domain to enable better security and data protection.
Write sequence to IP must follow a write handshake protocol:
Whenever the host wants to write to one of the IP registers, he must first assert CREQ field. Whenever the NFC will be ready to accept the write, he will assert CACK. Now the write sequence can go on. Upon completion of all write sequences, the host must clear the CREQ field.

The purpose of the handshake mechanism, is to prevent a situation where control registers are being changed during an operation and by that causing an unexpected behavior.

45.8.7 I/O Pins Sharing

The NAND Flash Controller has logic that allows it to share I/O pins with pins of another memory controller.

The arbitration between the NFC and the other memory has hard priority favouring the other memory. Since NFC's accesses are long, when the other memory requests the bus, the NFC will always halt its operation as soon as possible based on its internal state and the other memory will be granted. The only operations that will not halt in the middle are relatively short ones, such as command phase, address phase or spare area read. The host must take into account, that spare area read is the longest operation of the above and can take up to 32 Flash-clock cycles (for 8bit memory in non-symmetric mode).

This hard decision arbitration contains a dead-lock counter in favour of the NFC. The counter counts fast clocks cycles (aclk) in which the NFC requests the bus, but not being granted. If the counter reaches a certain value, it will cause the arbitration to favour the NFC over the other memory. Meaning, that if the other memory will deassert its request it won't get it again until the NFC will be granted at least once. After that grant, arbitration returns to normal mode, meaning, that if the other memory requests the bus, the NFC will pause as soon as possible based on the previous description.

This priority based arbitration mechanism must be taken into account when sharing the I/O bus with another memory.

45.9 NFC Operation

This section describes the operation of the NFC. It is divided to the following sub-sections:

- Automatic operations (45.9.1/45-64).
- Atomic operations (45.9.2/45-67).
- Atomic operations sequence (45.9.3/45-75).
- ECC operation (45.9.4/45-78).
- Symmetric/Asymmetric operation (45.9.5/45-80).
- Delay line operation(45.9.6/45-82).
- Write Protection peraion (45.9.7/45-83).

45.9.1 Automatic Operations

The NFC offers the user a simplification in accessing the NAND device, by automatically performing common NAND sequences. All automatic operations require some kind of preparations from the host, before the automatic operation is performed. NFC informs the host about the completion of the operation.

45.9.1.1 Automatic program operation

Setting AUTO_PROG bit in LAUNCH_NFC register, results in NFC executing a full page program including a status read at the end (The status-read operation will be executed only in case that RBB_MODE=0). NFC will repeat this operation several times as defined in NUM_OF_ITERATIONS field in NFC_CONFIGURATION1 register.
Upon setting AUTO_PROG bit, NFC will immediately set dma_wr_req (For systems that are using SDMA for filling the NFC's RAM). After setting dma_wr_req NFC will start executing the program operation by sending a command defined in NAND_CMD0 register, followed by sending the address from the relevant address group (The address will be divided into several address phases as defined in NUM_OF_ADDRESS_PHASES register). After the address phases finished the NFC will start copying the data from the NFC's RAM to the NAND device (If the RAM contains enough data as defined by FMP), and then NFC will issue a program-confirm command defined in NAND_CMD1 register. NFC will monitor the Ready/Busy status of the NAND device, and will read the status of the device as soon as the device returns to ready state.
When NUM_OF_ITERATIONS is greater than 0 and NUM_OF_DEVICES is greater than 0 (meaning interleaved mode. More than 1 page should be programmed to several devices), then NFC will operate a little different:
- NFC will set the next dma_wr_req after the data-transfer phase finished (meaning the RAM can be overwritten)
- NFC will start issuing the next page before the previous device returns to idle (Thus hiding the busy time of the device).
The address that will be sent depends on ADD_OP.
NFC will not set dma_wr_req after the data transfer of the last page (As defined in NUM_OF_ITERATIONS).
Automatic program operation generates 2 types of interrupts:
- INT will be set when all NAND devices that were involved in the operation returned to idle (And their status is recorded inside the NFC).

- AUTO_PROG_DONE will be set when NFC detects that all pages have been copied to its internal RAM (Meaning that the SDMA can be configured with parameters for next set of pages to be programmed).

45.9.1.2 Automatic Read Operation

Setting AUTO_READ bit in LAUNCH_NFC register, results in NFC executing a full page read. NFC will repeat this operation several times as defined in NUM_OF_ITERATIONS field in NFC_CONFIGURATION1 register.
Upon setting AUTO_READ bit, NFC will start executing a page read operation from the NAND device: read-command followed by address phases and a read-confirm-command if needed. Then NFC will monitor the read/busy status of the device, and will start reading the data as soon as the device returns to ready state. NFC will read the status of the operation afterwords.
NFC will set dma_rd_req when it finishes reading a full page from the NAND device (After fixing ECC errors if there are any).
If NUM_OF_ITERATIONS is greater than 0, then NFC pauses after reading a page, and waits for the system to read out the data from NFC's RAM. NFC will continue to read the next page as soon as the system reads enough data as defined in FMP register.
INT will be set after NFC has read the status register of all NAND devices that were involved in the operation.

45.9.1.3 Automatic erase operation

Setting AUTO_ERASE bit in LAUNCH_NFC register, results in NFC executing a full block erase. NFC will repeat this operation several times as defined in NUM_OF_ITERATIONS field in NFC_CONFIGURATION1 register.
Upon setting AUTO_ERASE bit, NFC will start executing a block erase operation from the NAND device: Erase-command, followed by erase-address, followed by erase-confirm-command. Then NFC will monitor the read/busy status of the device, and will read the status of the device as soon as the device returns to ready.
INT will be set when all NAND devices that were involved in the operation returned to idle (And their status is recorded inside the NFC).

45.9.1.4 Automatic copy-back operation

Setting AUTO_COPY_BACK0 bit in LAUNCH_NFC register, results in NFC executing a full copy-back operation.
NFC will send a command (defined in NAND_CMD0 register), then address phases (defined in the relevant address group), then another command phase (defined in NAND_CMD1 register), then NFC will monitor the ready/busy status of the NAND device, then another command phase (defined in NAND_CMD2 register), then address phases (defined in relevant address group), then last command phase (defined in NAND_CMD3 register), then again NFC will monitor the ready/busy status of the NAND device. Finally NFC will check the status of the operation.
INT will be set when the status of the NAND device is ready.
Setting AUTO_COPY_BACK1 will result in NFC executing AUTO_COPY_BACK0 operation to all NAND devices connected to it (with the same addresses to all the devices).

NFC will perform AUTO_COPY_BACK1 in an interleaved mode (meaning, that if a NAND device is getting busy, NFC will continue to the next device, and will get back to the first device as soon as it becomes ready). So, if the system wants to perform a copy-back operation with the same addresses to all the NAND devices, it is much more efficient to use AUTO_COPY_BACK1 then using AUTO_COPY_BACK0 several times.

45.9.1.5 Automatic status-read operation

Setting AUTO_STAT bit in LAUNCH_NFC register, results in NFC executing a full status read operation to the NAND device.
NFC will send status read command (defined in ST_CMD in NFC_CONFIGURATION2 register), then it will read the status of the device and store it in NF_STATUS field in NFC_CONFIGURATION1 register.

45.9.2 Atomic Operations

45.9.2.1 Preset Operation

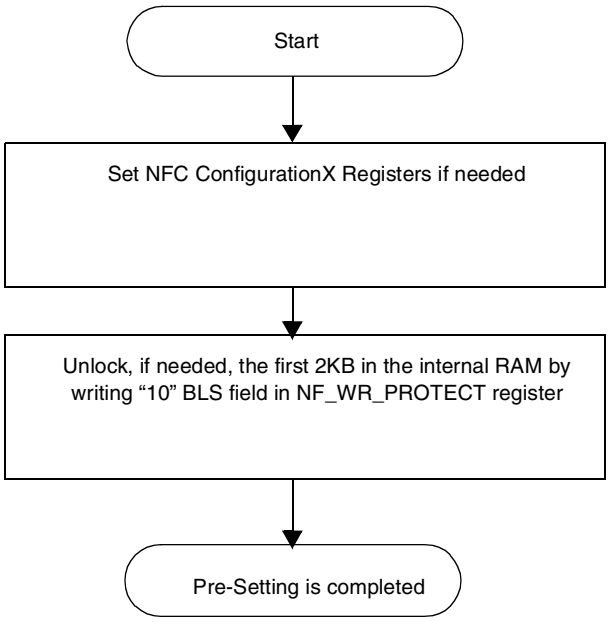


Figure 45-38. Flow Chart of Reset Operation

45.9.2.2 NAND Flash Atomic Command Input Operation

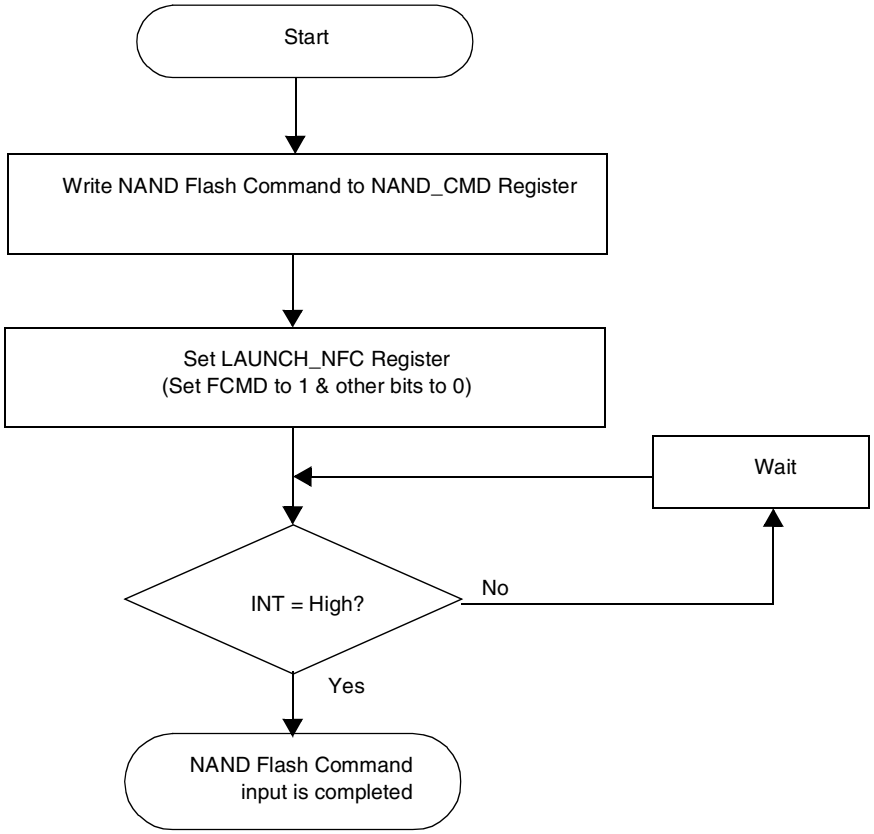


Figure 45-39. Flow Chart of NAND Flash Command Input Operation

45.9.2.3 NAND Flash Atomic Address Input Operation

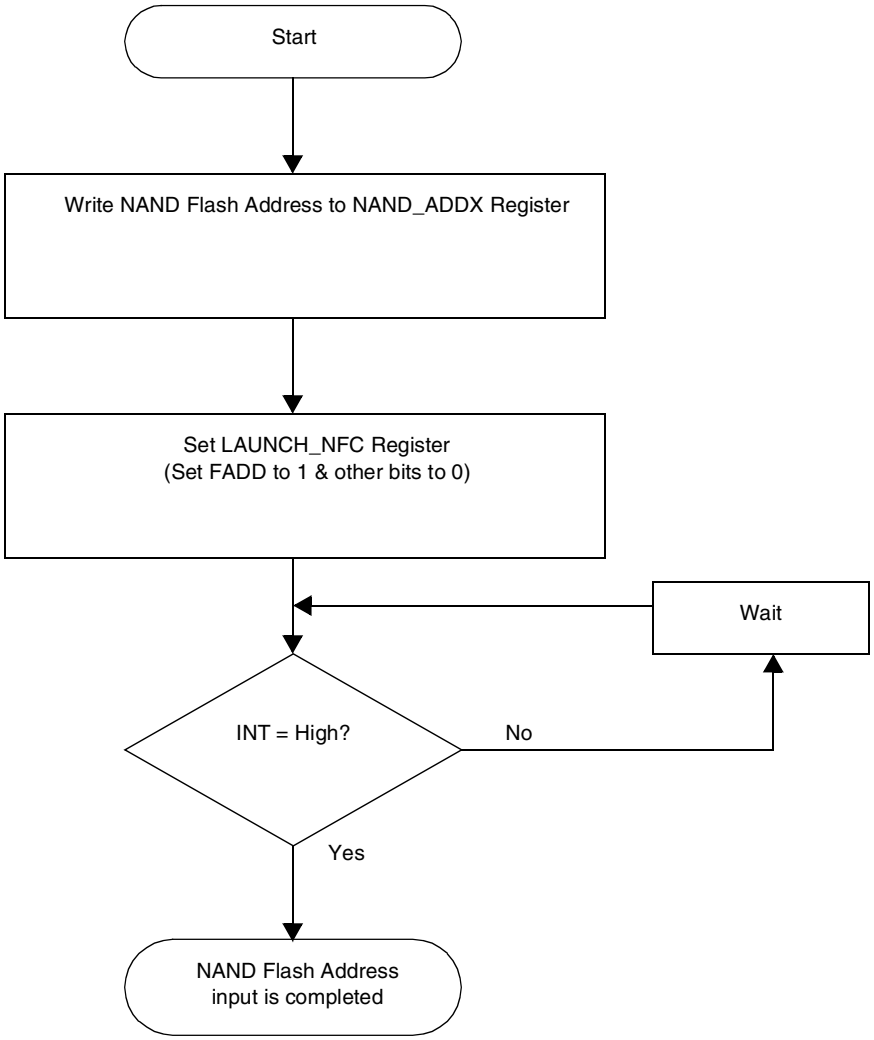


Figure 45-40. Flow Chart of NAND Flash Address Input Operation

45.9.2.4 NAND Flash Atomic Data Input Operation

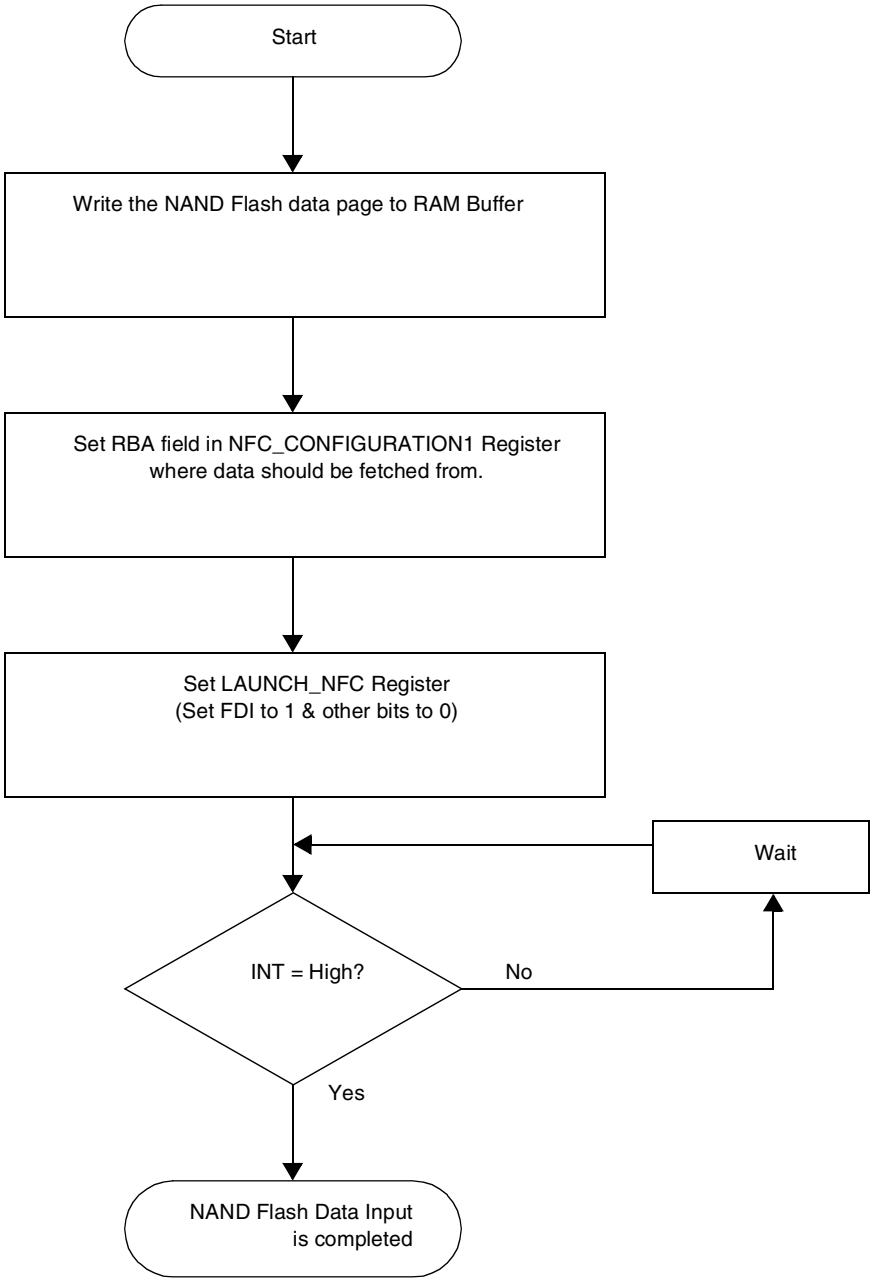


Figure 45-41. Flow Chart of NAND Flash Data Input Operation

45.9.2.5 NAND Flash Atomic Data Output Operation

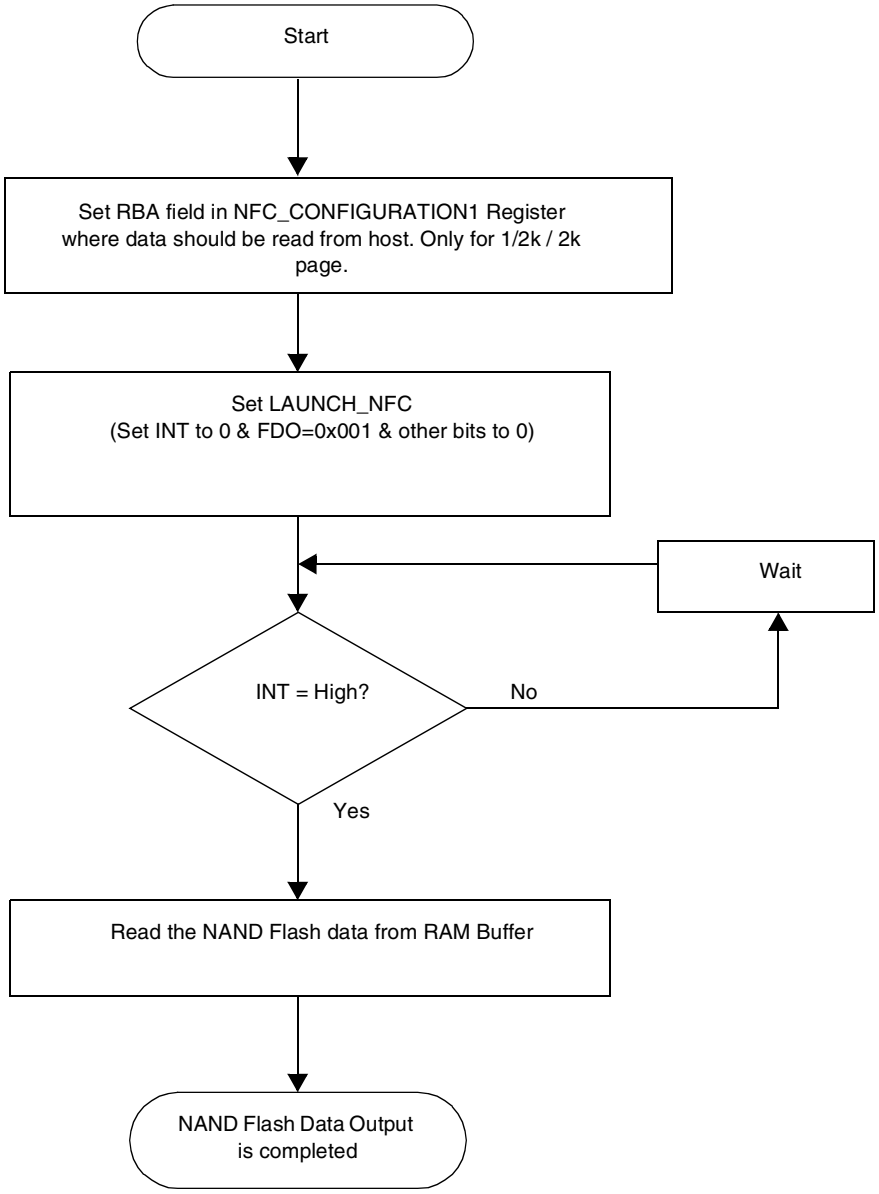


Figure 45-42. Flow Chart of NAND Flash Data Output Operation

45.9.2.6 Read NAND Flash Atomic ID Read Operation

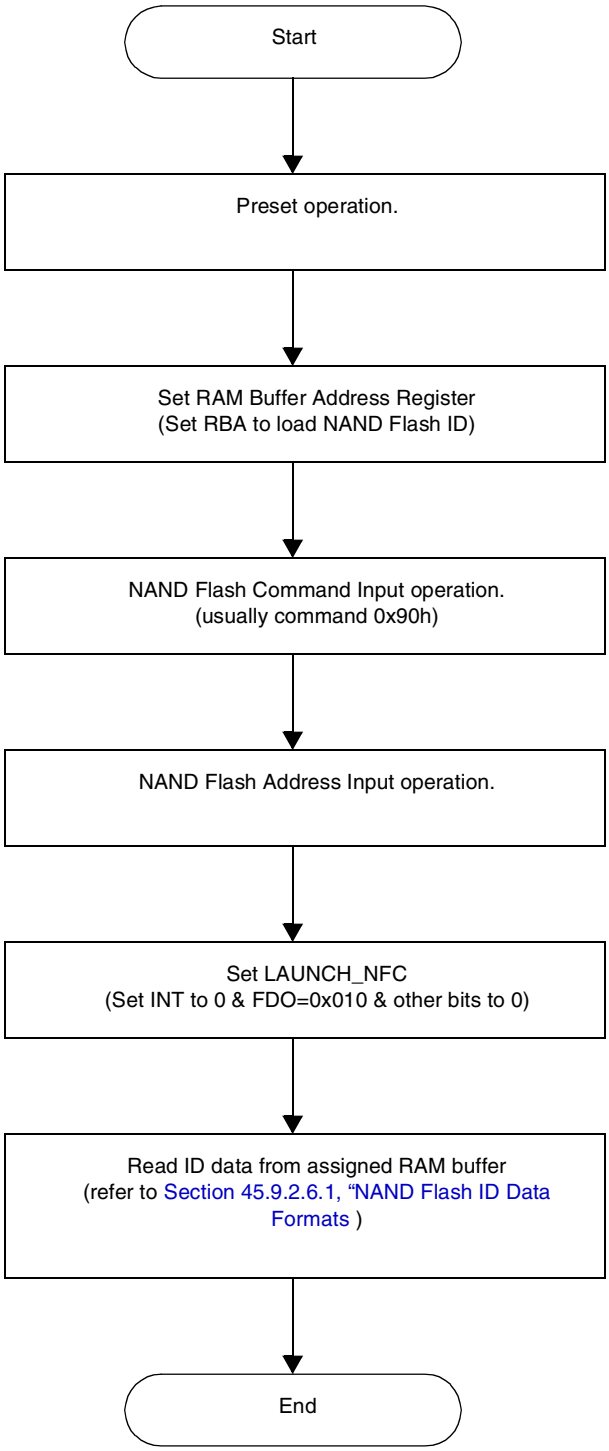


Figure 45-43. Flow Chart of Read NAND Flash ID Operation

45.9.2.6.1 NAND Flash ID Data Formats

The format of NAND Flash ID data stored in the RAM buffer (for x8 NAND Flash) is shown in Figure 45-44

RAM Buffer of RBA address				1st half-word				2nd half-word				3rd half-word				
				1st byte of ID		2nd byte of ID		3rd byte of ID		4th byte of ID		5th byte of ID		6th byte of ID		
				LSB			MSB									

Figure 45-44. NAND Flash ID Data Format (x8)

The format of NAND Flash ID data stored in the RAM buffer (for x16 NAND Flash) is shown in Figure 45-45.

RAM Buffer of RBA address				1st half-word				2nd half-word				3rd half-word				
				1st byte of ID		XXh		2nd byte of ID		XXh		3rd byte of ID		XXh		
				LSB			MSB									

RAM Buffer of RBA address				4th half-word				5th half-word				6th half-word				
				4th byte of ID		XXh		5th byte of ID		XXh		6th byte of ID		XXh		
				LSB			MSB									

Figure 45-45. NAND Flash ID Data Format (x16)

45.9.2.7 NAND Flash Atomic Status Read Operation

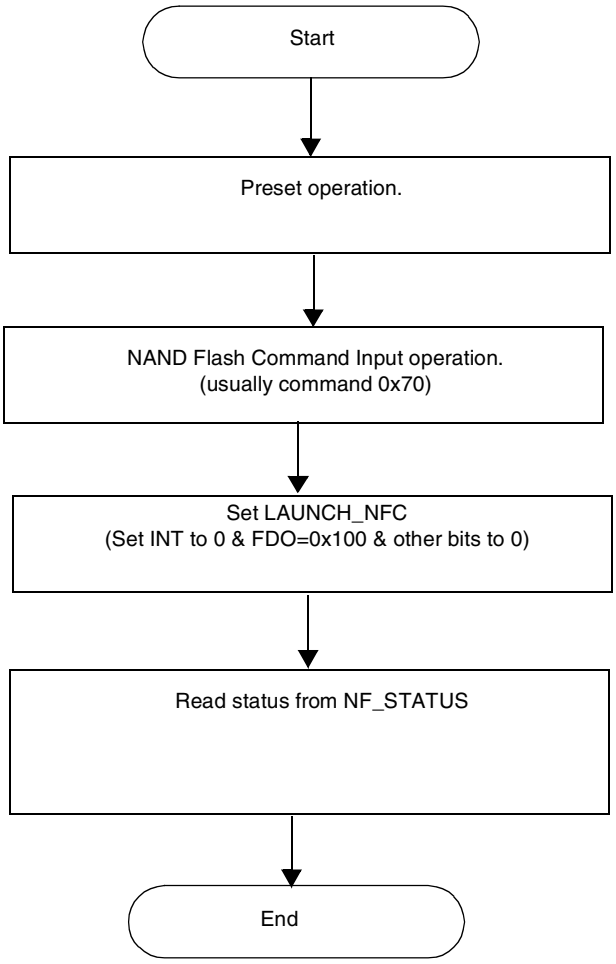


Figure 45-46. Flow Chart of Read NAND Flash Status Operation

45.9.3 Atomic Operations Sequence

45.9.3.1 Atomic Read Sequence Operation

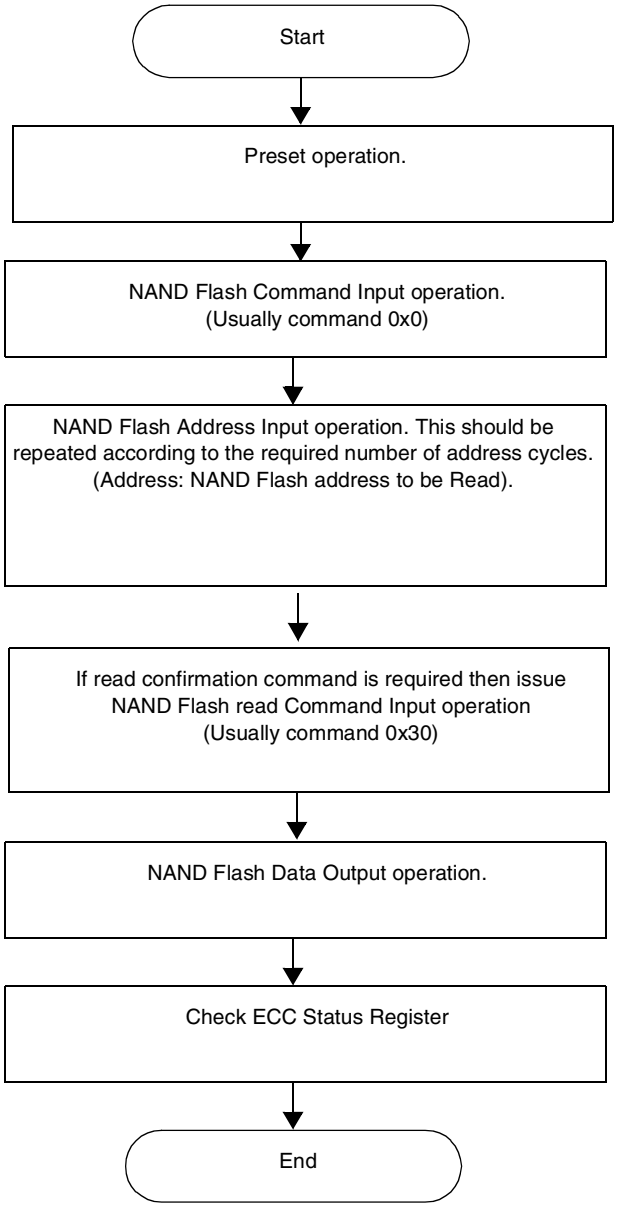


Figure 45-47. Flow Chart of Read NAND Flash Data Operation

45.9.3.2 Atomic Program Sequence Operation

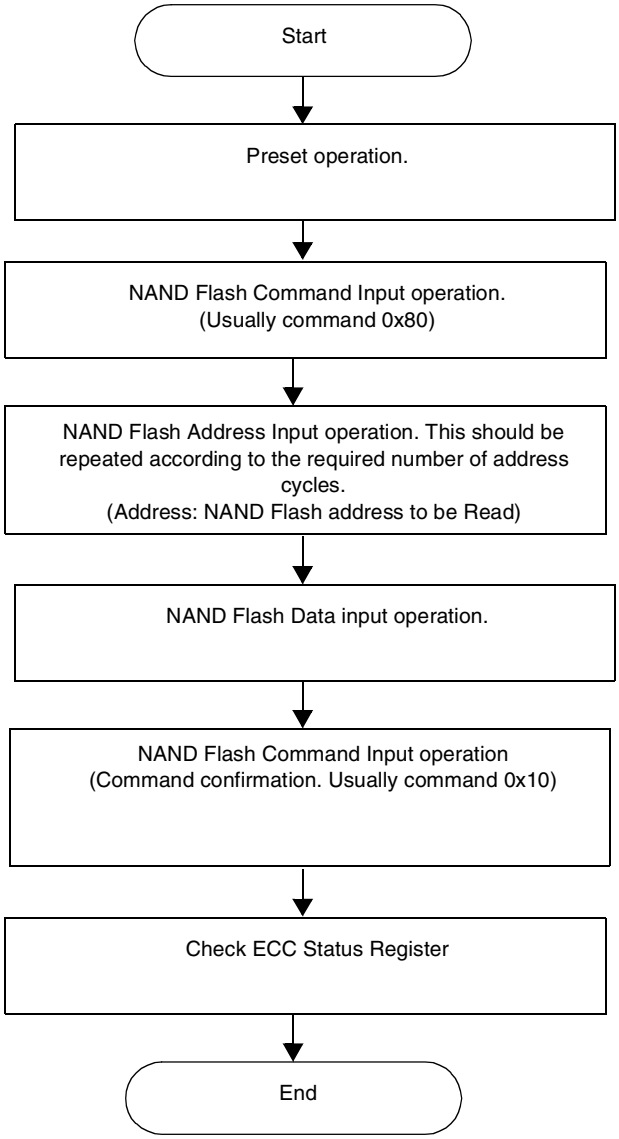


Figure 45-48. Flow Chart of Program NAND Flash Data Operation

45.9.3.3 Atomic Erase Sequence Operation

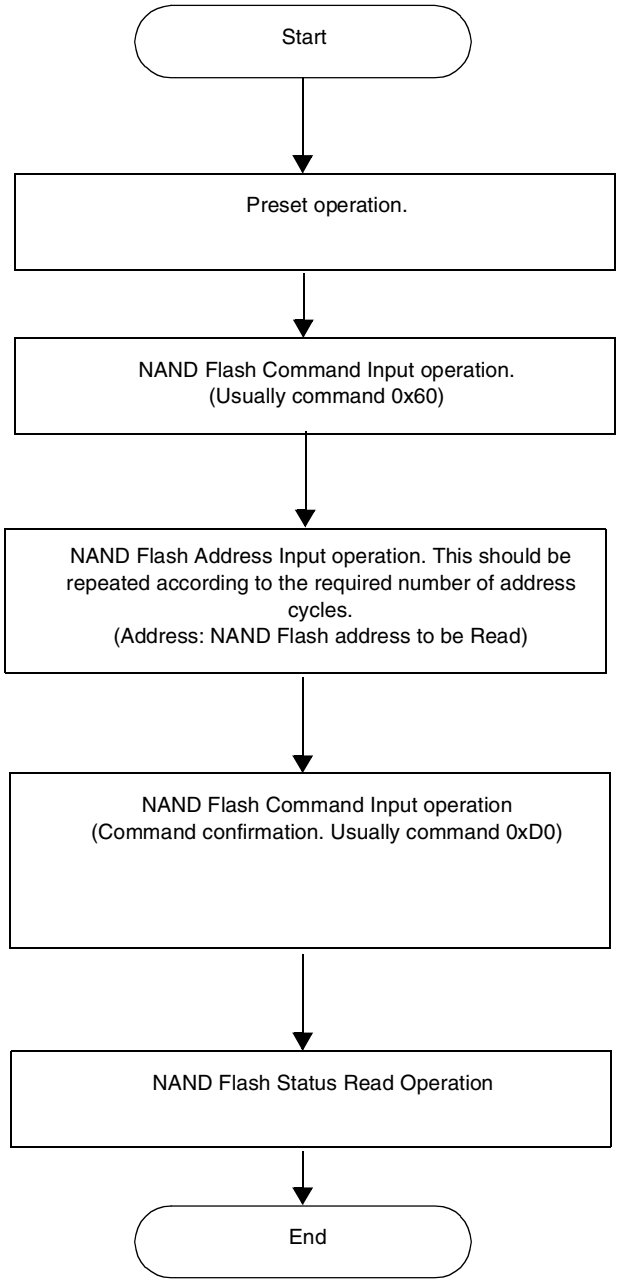


Figure 45-49. Flow Chart of Erase NAND Flash Operation

45.9.4 ECC Operation

45.9.4.1 ECC Normal Operation

The BCH ECC engine is responsible for detection and correction of up to 4/8 bits in each 528/538 byte section.
NFC divides the page into data sections of 1/2KB + spare area buffer. The ECC algorithm protects each data section and the associated spare area buffer.
When the NFC accesses the NAND Flash device for Program operation, it generates ECC code (8/14 bytes for each 528/538 bytes). These 8/14 bytes of ECC code are calculated based on the generator polynomial g(x) = x^13 + x^4 + x^3 + x + 1
When the NFC accesses the NAND Flash device for a Read operation, it performs a BCH decoding algorithm, and indicates how many bit errors were detected and corrected.
After a Read operation, the host can determine how many bit errors were found per 528/538 bytes by reading the status register (ECC_STATUS_RESULT). If the indication is uncorrectable error then it means that the number of bit errors that were detected (N) exceed the number of bit errors that can be corrected (T) according to ECC_MODE. Due to an algorithm weakness there might be a decoding error with a very low probability that the NFC will indicate that T number of bits error were detected and corrected instead of uncorrectable errors
Table 45-46 shows the ECC code assignment of the NAND Flash spare area.
Upon program, the BCH ECC bytes are written directly to the NAND Flash device (and are not written back to the internal RAM). This means that if a user wants to read the ECC that was generated, he must read it from the NAND Flash device spare area.

45.9.4.2 ECC Bypass Operation

In ECC bypass operation, the spare area is copied from NFC internal Ram Buffer to the nandflash device during program operation. During read operation the ECC detect-fix mechanism will not work as well as status register will not be updated.

Table 45-46. ECC Code/Result Readability

operation	Read operation		Program operation		
	ECC Code from spare area buffer	ECC status register	ECC Code from spare area buffer	ECC status register	ECC content in Nand Flash device
ECC operation	ECC code copied from nandflash device spare area	Valid	Invalid (old data [1])	-	ECC code genrated by the NFC
ECC bypass	ECC code copied from nandflash device spare area	Valid	Invalid (old data)	-	Data from spare area of NFC internal Ram

[1]Old data: ECC code is not updated to spare buffer, so ECC code placement of spare buffer remains old data.

45.9.4.3 How to Operate the ECC

In order to generate ECC and carry out the correction by the NFC,

— Program with ECC operation / Read with ECC operation

In order to generate ECC by the NFC and carry out the correction by the host,

— Program with ECC operation / Read without ECC operation

45.9.5 Symmetric/Asymmetric Mode Operation

In the default state of the NFC 1 flash clock cycle is used for toggeling RE#/WE#.
In order not to work with lower frequency of flash clock, the SYM bit in NFC_CONFIGURATION2 register should be set to “0” and that will change the WE# and RE# period during read or program to be 2 flash clock cycles instead of 1.
Setting SYM bit also changes the duty cycle of RE# to be ~50% during read operation.
Latching the data during a read operation is done 1 flash_clk cycle + 1 aclk cycle after negedge of RE#.
The user can adjust the c‘apturing timepoint by setting values to NFC_DEL_LINE register.

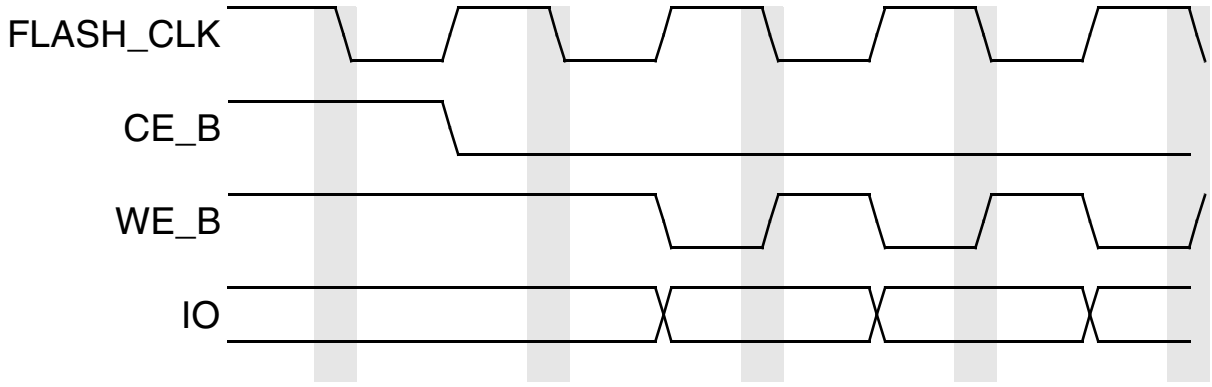


Figure 45-50. One Flash Clock Cycle Per Data Input (SYM bit =1)

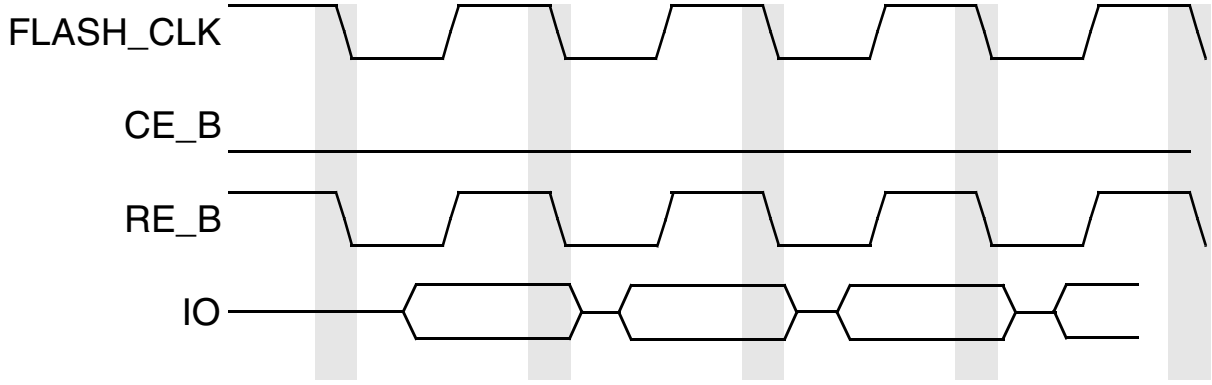


Figure 45-51. One Flash Clock Cycle Per Data Output (SYM bit =1)

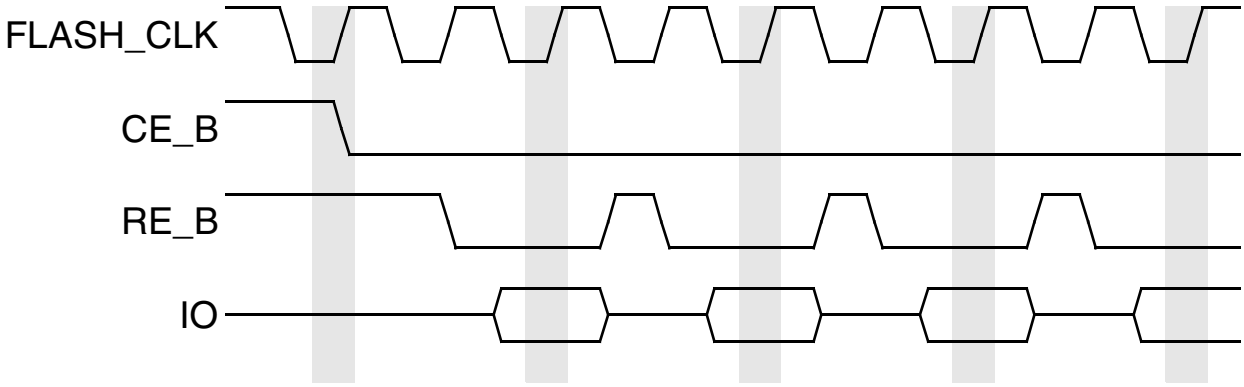


Figure 45-52. Two Flash Clock Cycles Per Data Output (SYM bit =0)

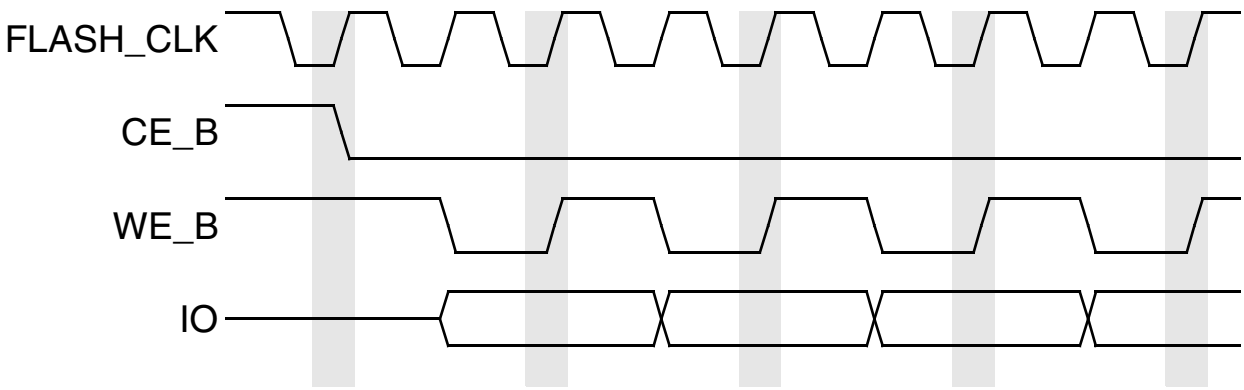


Figure 45-53. Two Flash Clock Cycles Per Data Input (SYM bit =0)

45.9.6 Delay Line Operation

NAND data-read path is a very long path. It Starts with RE_B negedge, then goes to the I/O PADs, through the board to the NAND device, which outputs the DATA after tREA [ns], then back through the I/O PADs until being latched by the NFC.

NFC offers a way of adjusting the data-latching timepoint, so that long read-data path will not require a degredation in frequency (And in performance).

The NFC generates an internal RE_B signal which is 1 flash_clk delayed from the RE_B that goes to the NAND device. This delayed RE_B is then passes through a delay-line (composed of 4 delay line units) which delays it by a configurable time measured by aclk fractions and up to 1/4 aclk cycle per delay-line unit. The delay-line output is used to latch the data.

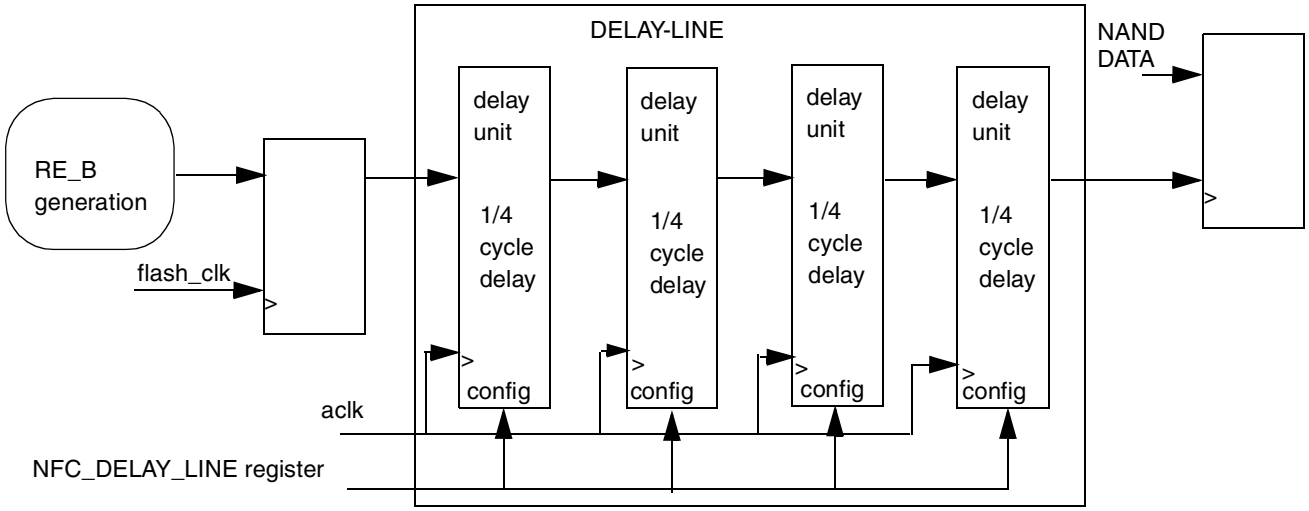


Figure 45-54. Delay Line Operation

45.9.7 Write Protection Operation

The NFC offers a software write protection feature, and a hardware write protection feature. Both are described in this section.

45.9.7.1 Write Protection for RAM Buffer (LSB 2KB)

The NFC offers a software write protection feature for the first 4 sections (main + spare area data) of the RAM buffer, which protects RAM buffer data. This write protection is carried out by setting the WPC bit of the NF_WR_Prot register.

The default state is locked state, and the first 4 sections go to this state after a cold or warm reset.

Write protection availability for main/spare memory regions in the RAM buffer are described on Table 45-7. A state diagram of RAM buffer write protection is shown in Figure 45-55

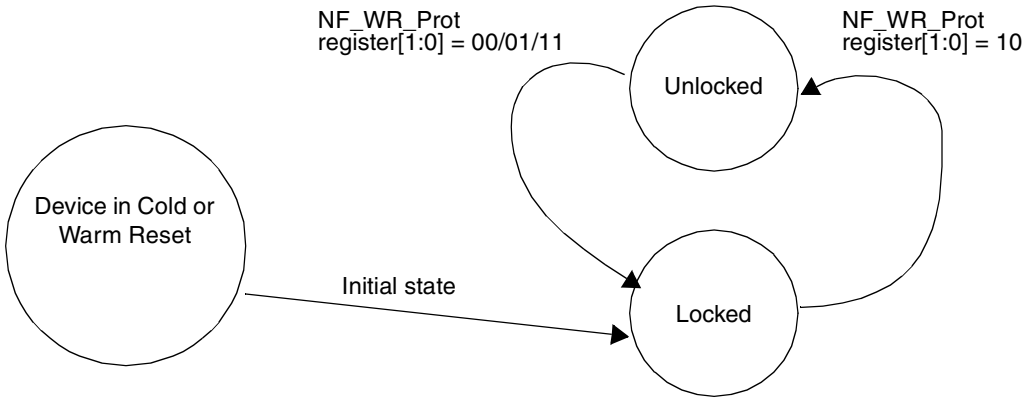


Figure 45-55. State Diagram of RAM Buffer Write Protection

45.9.7.2 Write Protection Modes

NFC offers both hardware and software Write Protection options for the NAND Flash device. The software Write Protection feature is used by executing the LOCK BLOCK command or LOCK-TIGHT BLOCK command, and the hardware Write Protection feature is used by executing a cold or warm reset. The WP signal is asserted only upon POR.

45.9.7.3 Write Protection Commands

There are two write protection states: **Locked**, and **Lock-Tight**.

- **Locked** state means that memory block in question is write protected (it cannot be written to), but the UNLOCK command can “un”-lock it. Useful for frequently changed memory blocks.
- **Lock-Tight** state is a higher level of protection, and means that the memory block in question is write protected, but the UNLOCK command cannot unlock it. Useful for memory blocks whose contents are rarely changed.
- If a user tries to modify a locked or locked-tight block, then the operation would not be carried to the Nand device and the user will not be notified about it.

The followings summarizes the locking functionality.

- — - All blocks power-up in a locked state. The UNLOCK command can unlock these blocks.
- — - The LOCK-TIGHT BLOCK command locks blocks and prevents it (them) from being unlocked.
  - – *Lock-Tight state can be reverted to locked state only when Cold/Warm reset is executed.*
- — - Writing to the unlock start/end address registers (Unlock_Start_Blk_Add and Unlock_End_Blk_Add) while the NFC is in the **Lock-Tight** state does not affect the unlock address.

45.9.7.4 Write Protection Status

The current Write Protection status of the NFC can be read in NAND Flash Write Protection status register (NAND_Flash_WR_Pr_St). There are three bits: US,LS, and LTS, which are not cleared by hot reset. These Write Protection status bits are updated as soon as the Write Protection command is entered.

Figure 45-56 shows a state diagram for the write protection of the NFC.

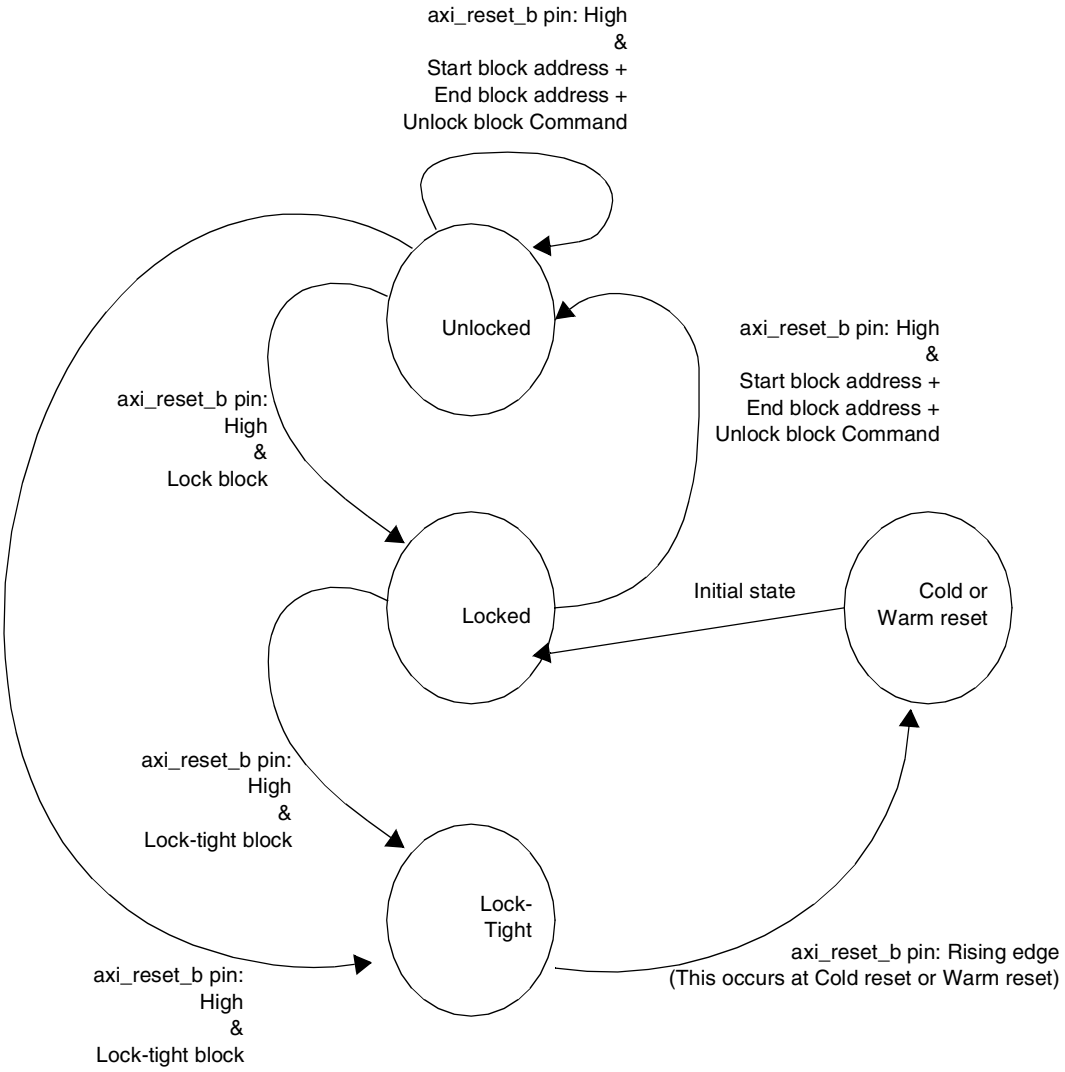


Figure 45-56. State diagram of NAND Flash Write Protection

45.9.7.4.1 Lock Sequence

The following describes the “lock” sequence:

1. Command Sequence: LOCK BLOCK Command (02h)
2. All blocks default to locked after initial Cold reset or Warm reset
3. Locking some of the blocks is not available; all memory blocks are locked upon reset
4. Unlocked memory blocks can be locked by using the LOCK BLOCK command. The status of a locked memory block can be changed to “unlocked” or “lock-tight” using the appropriate software commands.

45.9.7.4.2 Unlock Sequence

The following describes the “unlock” sequence:

1. Command Sequence: Start block address + End block address + UNLOCK BLOCK Command(04h)
2. Unlocked blocks can be programmed or erased
3. The status of an unlocked block can be changed to *locked* or *lock-tight* using the appropriate software commands
4. Only one sequential area can be released to unlocked state from locked state; Unlocking multi areas is not available

45.9.7.4.3 Lock-tight Sequence

The following describes the “lock-tight” sequence:

1. Only locked blocks can be “locked-tight” by the LOCK-TIGHT BLOCK command.
2. Command Sequence: LOCK-TIGHT BLOCK Command (01h)
3. Unlocking multi area is not available
4. Lock-tight blocks revert to the locked state at Cold/Warm reset.

45.9.8 Delay Line Operation

45.9.8.1 Delay line background

When a read access is issued towards the NAND flash device then it takes a relatively long time from the assertion of the RE_B till the read data is latched by the NFC. The read data path starts with the deassertion of the RE_B (read enable) signal through the propogation in the I/O pads and till the RE_B port of the NAND flash device on the board. When the NAND flash device detects the assertion of the RE_B signal it drives the associated read data back after Trea[ns] through the I/O pads and till being latched by the NFC. The NAND flash device is committed to keep driving the data for Trhoh [ns] after the posedge of RE_B. The delay line is used to compensate the long delay of the read data path and to adjust the sampling edge of the read data without any performance or frequency degradation.

The NFC generates an internal RE_B signal which is delayed by 1 flash clock cycle from the RE_B that is driven to the NAND device. The delayed RE_B passes through a delay line and the output of the delay line is used to latch the read data.

The delay line is composed from 4 delay line units. Each unit can delay the RE_B signal by a configurable time measured by axi slow clock (fast clock of the NFC) fractions and up to 1/2 axi slow clock cycle. So the delay line can shift the RE_B signal by total of 2 axi slow clock cycles. Each fraction is 1/256 part of axi slow clock cycle and is configured in NFC_DELAY_LINE[NFC_ABS_DEL] register. This register is associated with a single delay line unit. By default the NFC latches the data after a fixed delay of 1 flash clock cycle + 1 axi slow clock cycle.

Figure 45-57 shows the read data path and Figure 45-58 shows the associated timing diagram.

45.9.8.2 Delay line and flash clock settings

The following will show how to set the flash clock cycle (2*Trp) and configure the desired value of the delay line. The calculations are based on parameters which depends on the NAND flash device, Freescale controller and the operating conditions.

- Assume the internal delay of the RE_B + the internal delay of the read data is D1[ns]
- Assume the board delay of the RE_B + board delay of the read data is D2[ns]
- Assume the delay of the RE_B caused by the delay line is Ddel[ns]. Note that the delay line has a minimal delay even if configured to zero delay
- Assume Trea[ns] is RE_B access time
- Assume Trp[ns] is RE_B pulse width
- Assume Trhoh[ns] is read data hold time from RE_B high

Note: The parameter D1 can be obtained from the AC characteristics of the Freescale device.

In order to sample the read data on time it is required that:

- 2Trp +Ddel > Trea + D1 + D2
- Trp + Ddel < Trhoh + D1 + D2
- Trp > Trp min defined by the NAND flash AC characteristics.

For example if:

- D1 = 10ns
- D2 = 2ns
- Delay line is configured to zero delay and its minimal delay is Ddel=3ns
- Trea = 20ns
- Trhoh = 15ns
- Trp min defined by the device is 12ns
- axi slow clock cycle is 7.5ns

In that case the pulse width of flash clock should be 14.5ns < Trp < 24ns or in other words the cycle of flash clock should be 29ns < Trc < 48ns (i.e flash clock frequency is 20.8Mhz < F< 34.4Mhz)

In case it required to work with Trp which is 12ns (i.e the minimal allowed by the NAND device) then it is possible by using the delay line as following. Setting a delay of 14.5 - 12 = 2.5ns means that we should delay the RE_B signal by 2.5/7.5 = 1/3 cycle of slow axi clock, which is approximately 85 parts of 256, So it is needed to configure NFC_DELAY_LINE[NFC_ABS_DEL] = 8'd85

Note:
For the above example the allowed ratio between axi slow clock cycle and flash clock cycle are 1:4 ,1:5 or1:6. If the cycle of the flash clock is set to 37.5ns (i.e ratio 1:4) and the total delay of the delay line is configured to 1 axi slow clock cycle(as the default value) then the latching edge of the read data will be delayed by 1 axi slow clock cycle, which is 7.5ns and it isn't guaranteed that the read data will be latched correctly.

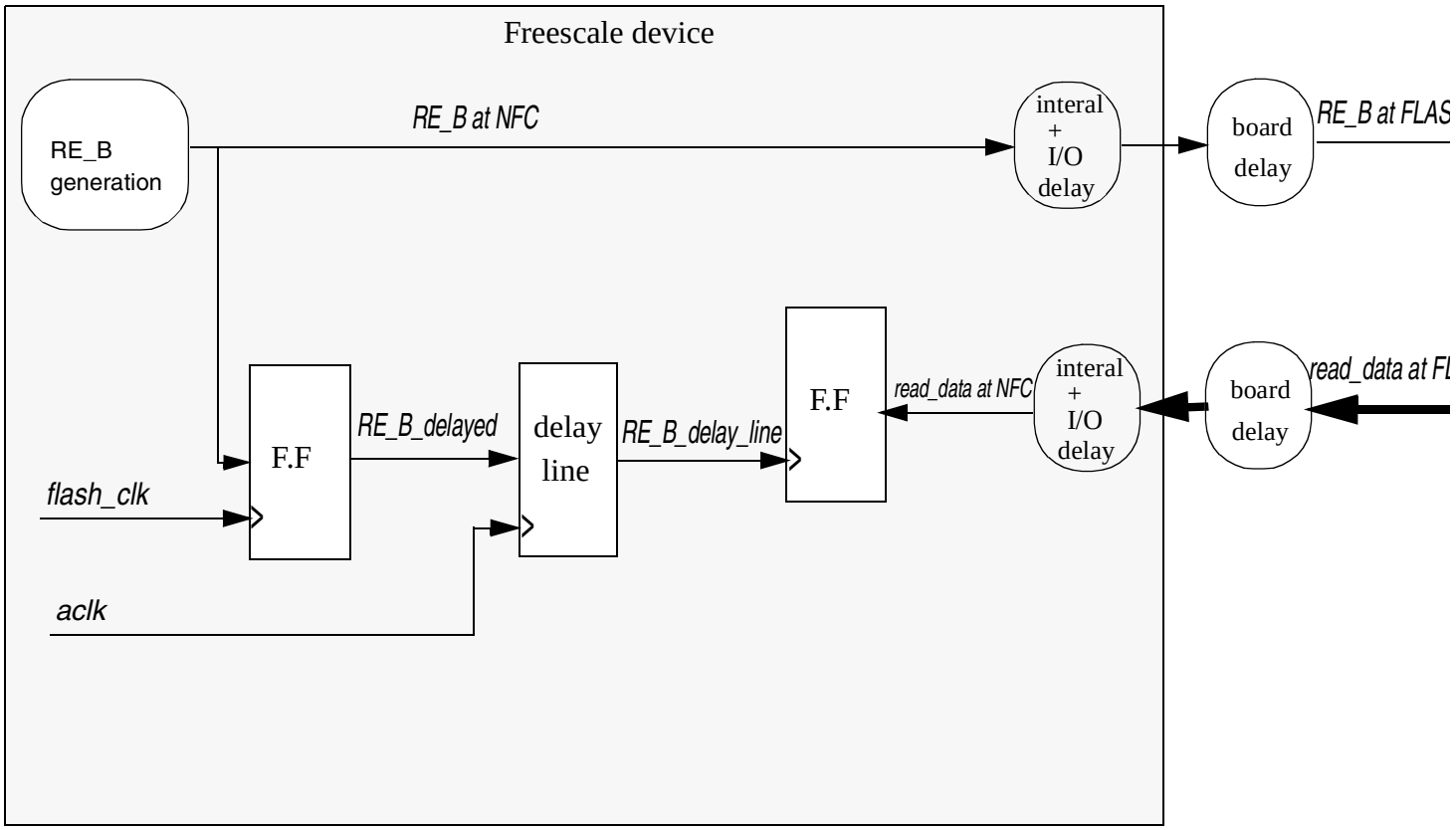


Figure 45-57. Read data path

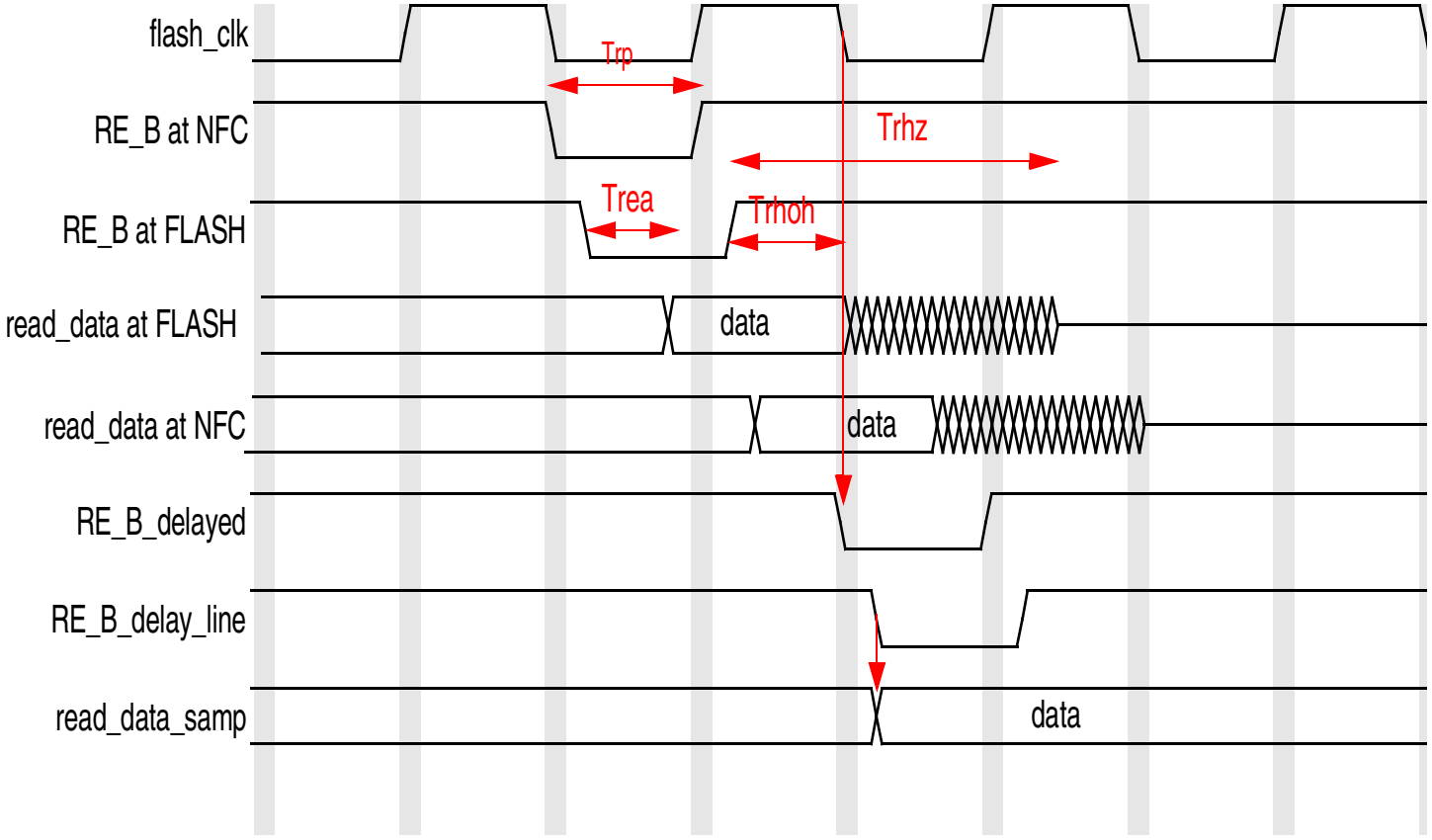


Figure 45-58. Read data path timing

Figure 45-59 shows the block diagram of the delay line

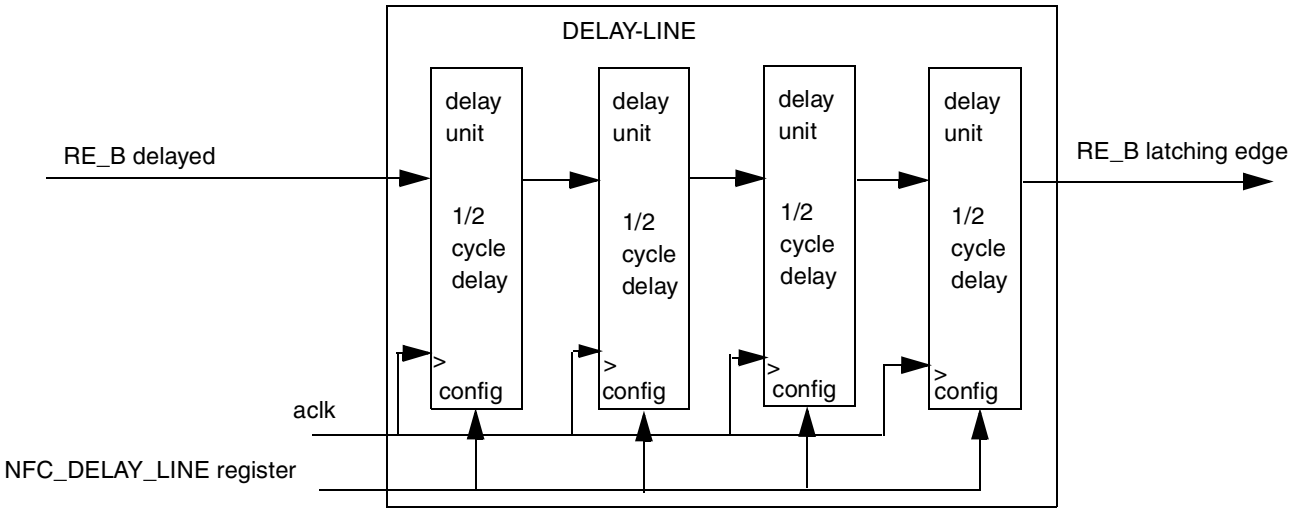


Figure 45-59. Delay Line block diagram

45.10 Memory Connectivity Examples

Figure 45-60 and Figure 45-61 shows connectivity to 8-bits and 16-bits NAND device respectively

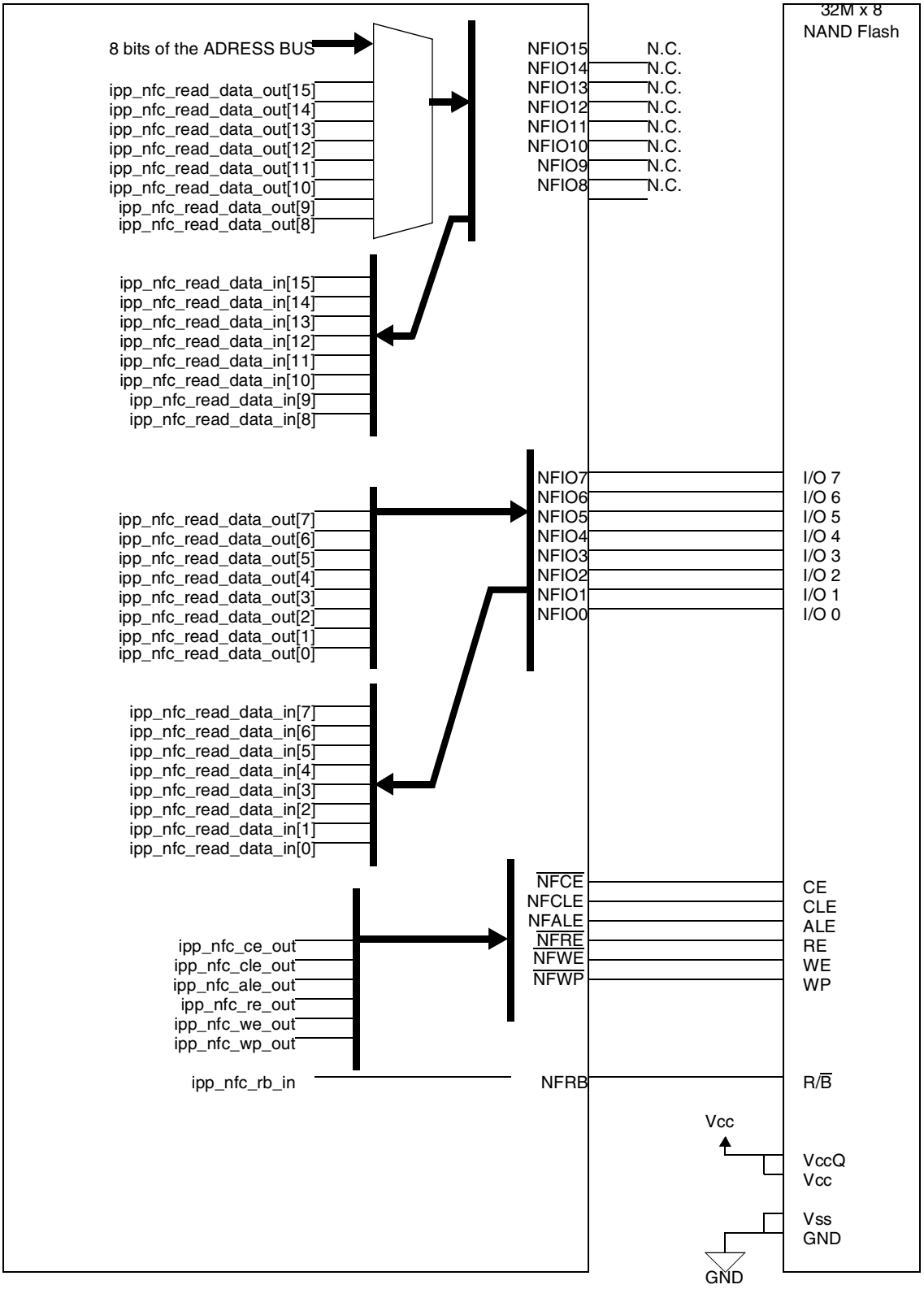


Figure 45-60. 8-bit NAND Flash Connectivity Diagram

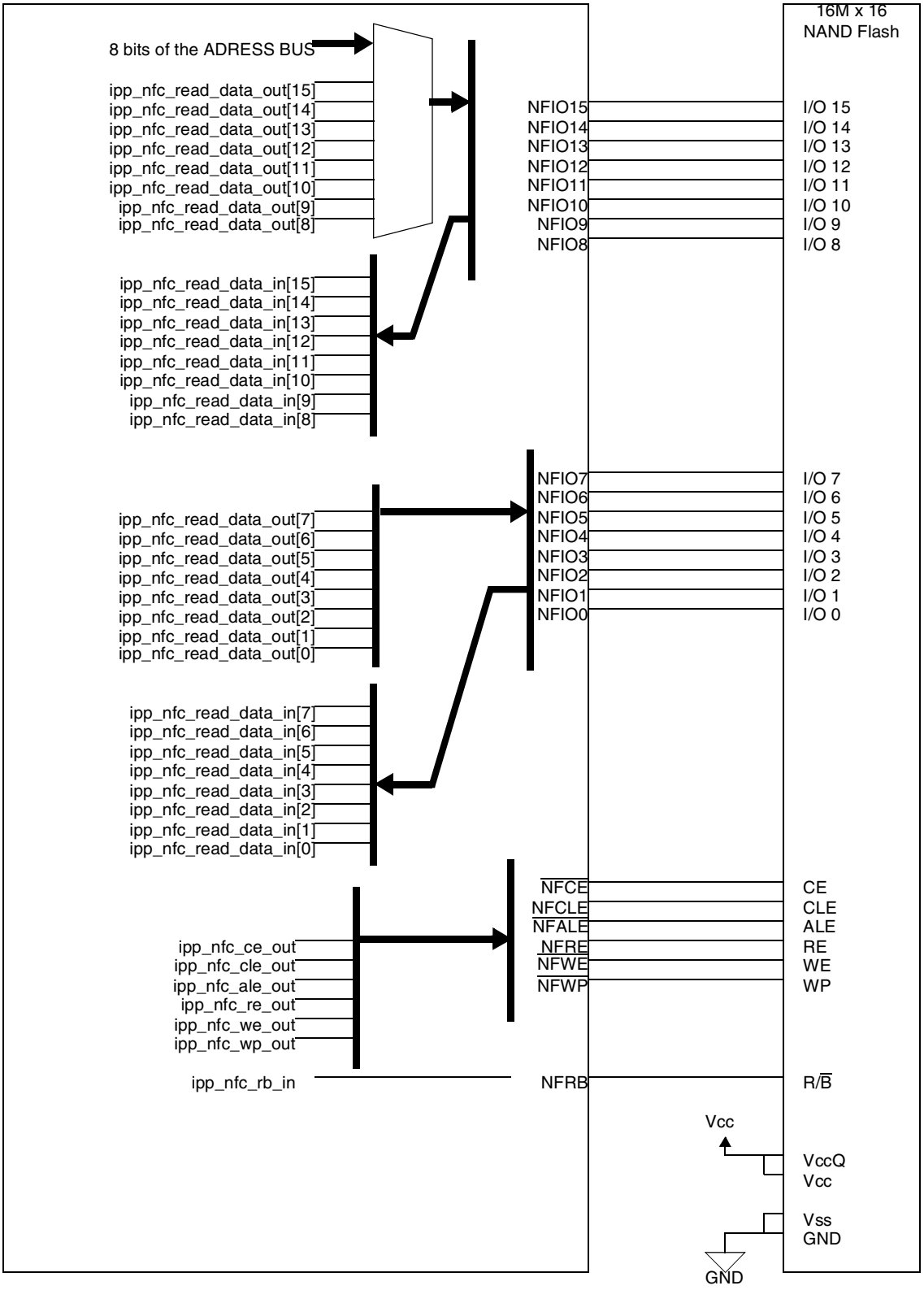


Figure 45-61. 16-bits NAND Flash Connectivity Diagram

Note:

- The NFC controller can support HS (High Speed) NAND Flash by supplying higher frequencies to the Flash Clock input.

45.11 Verified Nand Models

The Following Table lists the Nand models that where used for verifying the NFC, and are gauranteed to be supported.

Table 45-47. Verified Nand Models

Vendor	Part number	page size	bus width
Samsung	k9f1g08u0m	2K	x8
	k9f1g16u0m	2K	x16
	k9f5608q0c	1/2K	x8
	k9k8g08u0m	2K	x8
	k9f5616q0c	1/2K	x16
	k9gag08u0m	4K	x8
Toshiba	tc58dvg02a1fti0	1/2K	x8
	tc58dvm82f1ft00	1/2K	x16
	nfc_tc58nvg0s3aft05	2K	x8
	nfc_tc58nyg0d9bxgj5	2K	816
	nfc_tc58nvg4d1dtg00	4K	x8
Hynix	nfc_hy27ss08121a	1/2K	x8
	nfc_hy27ss16121a	1/2K	x16
	nfc_hy27uf082g2m	2K	x8
	nfc_hy27uf162g2m	2K	x16
ST	nfc_nand02gw3b	2K	x8
	nfc_nand04gw3b	2K	x8
	nfc_nand512w3a2	1/2K	x8
Micron	nfc_mt29f2g08aab	2K	x8
	nfc_mt29f2g16aab	2K	x16

Chapter 46
Parallel Advanced Technology Attachment (P-ATA)

The ATA block is an AT attachment host interface. Its main use is to interface with hard disc drives and optical disc drives. It interfaces with the ATA device over a number of ATA signals. See Figure 46-1 for the block diagram of the ATA interface.

It is possible to connect a bus buffer between the host side and the device side. In this case, the ata_buffer_en signal should be used to control the direction the buffer is driving to. If ata_buffer_en is high, it drives outward to the device. If ata_buffer_en is low, it drives inward to the host.

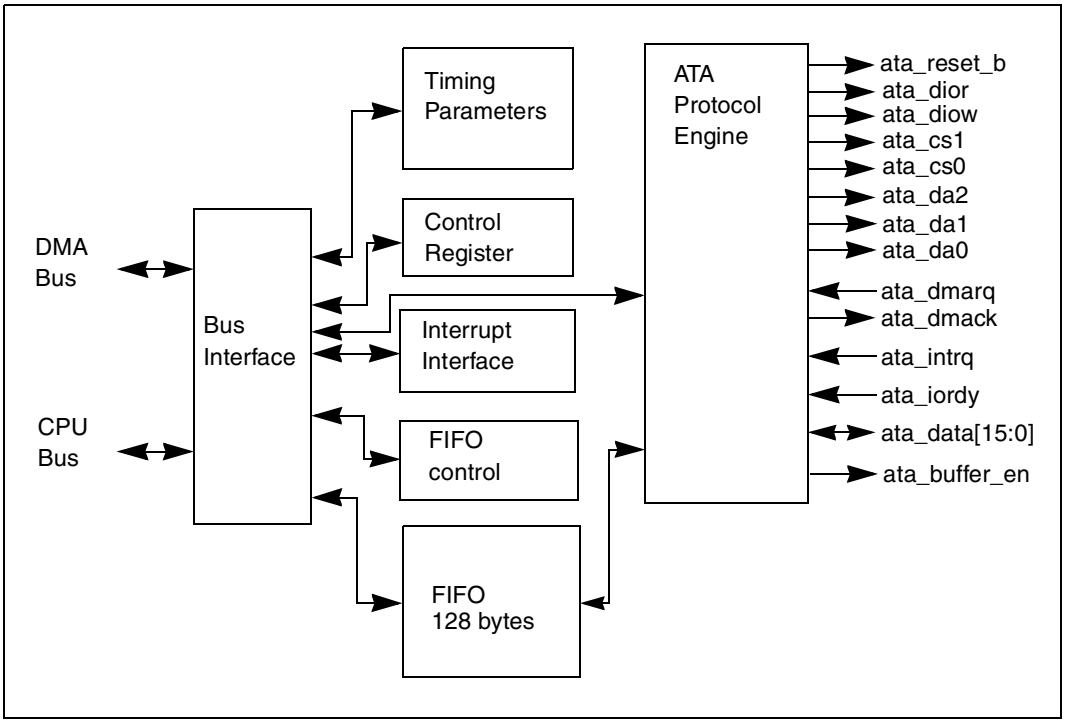


Figure 46-1. ATA interface Block Diagram.

46.1 Overview

The ATA block is a AT attachment host interface. Its main use is to interface with IDE hard disc drives and ATAPI optical disc drives. It interfaces with the ATA device over a number of ATA signals.

The ATA interface is compliant to the ATA-6 standard and supports the following protocols:

- PIO mode 0, 1, 2, 3, and 4
- Multiword DMA mode 0, 1 and 2
- Ultra DMA modes 0, 1, 2, 3 and 4 with bus clock of 50 MHz or higher
- Ultra DMA mode 5 with bus clock of 80 MHz or higher.

The ATA interface has 2 busses connected:

- One CPU bus for communication with the host processor
- One DMA bus for communication with the host DMA unit

All internal registers are visible from both busses, allowing smart DMA access to program the interface.

Before accessing the ATA bus, the host must program the timing parameters to be used on the ATA bus. The timing parameters control the timing on the ATA bus. Most timing parameters are programmable as a number of clock cycles (1 to 255). Some are implied.

After programming the timing parameters, there are two protocols that can be active at the same time on the ATA bus:

- First protocol. This protocol is a PIO mode access that can be performed at any time by the host CPU or the host smart DMA to the ATA bus. During PIO mode access, the incoming IP bus cycle is translated to an ATA bus cycle by the ATA protocol engine. The IP bus cycle is stalled until completion of the ATA bus cycle on read, or until putting the write data on the ATA bus on write. The PIO mode is a slow protocol, mainly intended to program the ATA disc drive, but also possible to use to transfer data to/from the disc drive. During PIO mode, the FIFO is not active.
- Second protocol. This protocol is the DMA mode access. DMA mode is started by the ATA interface after receiving a DMA request from the drive, and only if the ATA interface has been programmed to accept the DMA request. In DMA mode, either multiword DMA or ultra DMA protocol is used on the ATA bus. Once started, data transfer is organized between the ATA bus and the FIFO. Data transfer will pause to prevent FIFO overflow / FIFO underflow. Data transfer will resume when there is again space in the FIFO, or when the FIFO has been refilled. During DMA transfer, there is no direct interface between the ATA bus and the host IP or host DMA IP bus. Instead, the transfer takes place between the ATA bus and the FIFO; the FIFO informs the host DMA unit when it needs to be refilled or emptied. In this case, it sends an ALARM flag to the host DMA. When the host DMA receives the fifo_tx_alarm, it should write some data to the FIFO. (typically 32 bytes). When the host DMA receives the fifo_rcv_alarm, it should read some data from the FIFO (typically 32 bytes). The FIFO filling level at which the alarms are produced, is programmable. For completeness, there is a third alarm associated with the host DMA operation fifo_txfer_end_alarm. This alarm signals the end of the transfer, and requests the smart DMA to take steps to complete the transfer: transfer the bytes remaining in the FIFO to the host memory, and inform the host CPU the transfer is completed.

All transfers between FIFO and host IP or DMA IP bus are zero wait states transfer, so high-speed transfer between FIFO and DMA/host bus is possible.

When a PIO access is performed during a running DMA transfer, the DMA transfer will be paused, the PIO access done, and the DMA transfer will resume again.

46.1.1 Features

The ATA interface includes the following features:

- Programmable timing on the ATA bus. Works with wide range of bus frequencies.
- Compliant with ATA-6 specification
  - Supports PIO modes 0, 1, 2, 3 and 4
  - Supports multiword DMA modes 0, 1 and 2
  - Supports ultra DMA modes 0, 1, 2, 3 and 4 with bus clock of at least 50 MHz
  - Supports ultra DMA mode 5 with bus clock of at least 80 MHz
- Can be used with off-chip bus transceiver if pads are not compliant with ATA voltage levels
- 128-byte FIFO part of interface
- FIFO receive alarm, FIFO transmit alarm, and FIFO end of transmission alarm to DMA unit
- Zero-wait cycles transfer between DMA bus and FIFO allows fast FIFO reading/writing

46.1.2 Modes of Operation

The interface offers two operation modes that can be used together:

- PIO Mode

  An access to the ATA bus in PIO mode occurs whenever a ATA pio register is read or written by the host CPU or the host (smart) DMA unit. During a PIO transfer the incoming IP bus cycle is translated to an ATA PIO bus cycle by the ATA protocol engine. No buffering of data occurs, so the host CPU or host DMA cycle is stalled until the ATA bus read data is available on read, or is stalled until the IP bus data can we put on the ATA bus during write.

  PIO accesses can be done to the bus at any time, even during a running ATA DMA transfer. In this case, the DMA transfer is paused, the pio cycle is completed, and the DMA transfer is resumed.
- DMA Mode

  In DMA mode, data is transferred between the ATA bus and the FIFO. Two different DMA protocols are supported on the ATA bus: ultra DMA mode and multiword DMA mode. Selection is made using a control register bit.

  A DMA transfer will be started when DMA mode transfer has been enabled by writing some control bit, and when the drive connected to the ATA bus pulls its DMARQ line high.

  During an ATA bus DMA transfer, data is transferred between the ATA bus and the FIFO. The transfer will pause to avoid FIFO overflow and FIFO underflow.

  It is the task of the host CPU or the host smart DMA unit to read data or write data to the FIFO to keep the transfer going. Normal set-up is that the host (smart) DMA unit takes on this task. For this purpose, the fifo_rcv_alarm and fifo_tx_alarm signals are sent to the host DMA unit. fifo_rcv_alarm informs the host DMA unit that there is at least 1 packet of data waiting in the FIFO to be read by the host DMA. Whenever this signal is high, the host DMA should transfer one packet of data from the FIFO to the main memory. Typical packet size is 32 bytes (8 long words), but other packet sizes can be handled too. fifo_tx_alarm informs the host DMA unit that there is space for at

least 1 packet to be written by the host DMA. Whenever this signal is high, the host DMA should transfer one packet of data from main memory to the FIFO. Typical packet size is 32 bytes (8 long words), but other packet sizes can be handled too.

46.2 External Signal Description

See Table 46-1 for the list of signals entering and existing this module to peripherals within the chip.

Table 46-1. Signal Properties

Name	Port	Function	Reset State	Type
ata_reset_b	—	ATA bus reset signal. Active low. If active, ata device is reset[1]	0	out
ata_dior	—	ATA bus read strobe	1	out
ata_diow	—	ATA bus write strobe	1	out
ata_cs1	—	ATA bus chip select 1	1	out
ata_cs0	—	ATA bus chip select 0	1	out
ata_da2	—	ATA bus address line 2	0	out
ata_da1	—	ATA bus address line 1	0	out
ata_da0	—	ATA bus address line 0	0	out
ata_dmarq	—	ATA bus DMA request	—	in
ata_dmack	—	ATA bus DMA acknowledge	1	out
ata_intrq	—	ATA bus interrupt request	—	in
ata_iordy	—	ATA bus iordy	—	out
ata_data[15:0]	—	ATA data bus (little-endian)	Hi-z	tri-state in-out
ata_buffer_en	—	Buffer enable for external bus transceiver[2]	0	out

[1]This signal is a standard ATA bus signal. It conforms with the ATA specification.
[2]It is optional to put a 74xxx245 bus transceiver between the host side of the data bus and the device side of the data bus. If the transceiver is used, its enable should be tied low (always enable), and its direction pin should be tied to ata_buffer_en, in such a way that it drives from host to device when ata_buffer_en is high, and drives from device to host when ata_buffer_en is low.

46.2.1 Detailed Signal Descriptions

For a detailed description of the ATA bus signal, refer to the ATA-6 specification.

46.2.1.1 ata_reset_b (out)

This signal is the ATA reset signal. When low, the ATA bus is in reset state. When high, no reset. The ATA bus is in reset whenever the appropriate bit in the control register is cleared. After system reset, the ATA bus is in reset.

46.2.1.2 ata_dior (out)

This signal correspond to ATA signal DIOR. During PIO and multiword DMA transfer, function is read strobe. During ultra DMA in burst, function is HDMARDY. During ultra DMA out burst, function is host strobe.

46.2.1.3 ata_diow (out)

This signal corresponds to ATA signal DIOW. During PIO and multiword DMA transfer, function is write strobe. During ultra DMA burst, function is STOP, signalling whenever host wants to terminate running ultra DMA transfer.

46.2.1.4 ata_cs0, ata_cs1, ata_da2, ata_da1, ata_da0 (out)

These signals are the address group of the ATA bus. ata_cs0, ata_cs1 are the chip selects; ata_da2, ata_da1 and ata_da0 are the 3 address lines. All these five lines follow the same timing.

46.2.1.5 ata_dmarq (in)

This signal is the ATA bus device DMA request. Its pulled high by the device if it wants to transfer data using multiword DMA or ultra DMA mode

46.2.1.6 ata_dmack (out)

This signal is the ATA bus host DMA acknowledge. Its pulled low by the host when it grants the DMA request.

46.2.1.7 ata_intrq (in)

This signal is the ATA bus interrupt request. Its pulled high by the device whenever it wants to interrupt the host CPU.

46.2.1.8 ata_iordy (in)

This signal is the ATA bus IORDY line. It has the following functions:

- IORDY—active low wait during PIO cycles
- DDMARDY—active low device ready during ultra DMA out transfers
- DSTROBE—device strobe during ultra DMA in transfers

46.2.1.9 ata_data[15:0] (in/out–tristate)

This is the ATA data bus.

46.2.1.10 ata_buffer_en

This is a buffer direction control signal.

46.2.2 Electrical Spec on the ATA Bus, Bus Buffers

To meet electrical spec on the ATA bus, several requirements must be met. For a detailed description, refer to the ATA specification.

This electrical spec must be met for the pads used on the ATA I/Os if no bus buffers and bus transceivers are used.

Alternative is to use bus buffers. This is the only way to operate the ATA interface if 3.3 V or 5.0 V compatibility on the ATA bus is wanted, and no 3.3 V or 5.0 V tolerant pads on the device are available.

The use of bus buffers introduces delay on the bus and introduces skew between signal lines. These factors will make it difficult to operate the bus at the highest speed (UDMA-5) when bus buffers are used. If fast UDMA mode operation is needed, this may not be compatible with bus buffers.

Another area of attention is the slew rate limit imposed by the ATA specification on the ATA bus. According to this limit, any signal driven on the bus should have a slew rate between 0.4 and 1.2 V/ns with a 40 pF load. Not many vendors of bus buffers specify slew rate of the outgoing signals.

When bus buffers are used, the ata_data bus buffer is special. This is a bidirectional bus buffer, so a direction control signal is needed. This direction control signal is ata_buffer_en. When its high, the bus should drive from host to device. When its low, the bus should drive from device to host. Steering of the signal is such that contention on the host and device tristate busses is always avoided.

46.2.3 Timing on ATA bus

Timing on the ATA bus is explained in this section. It is also explained how to make sure the ATA interface meets timing. Timing is explained with timing figures and also equations are provided that need to be fulfilled for the host to meet timing.

46.2.3.1 Timing Parameters

In the timing equations, some timing parameters are used. These parameters depend on the implementation of the ATA interface on silicon, the bus buffer used, the cable delay and cable skew. Refer to Table 46-2 for the list of parameters used to specify the ATA timing.

Table 46-2. Timing Parameters

Name	Meaning	Controlled by
T	Bus clock period	clock generator
ti_ds	Set-up time ata_data to ata_iordy edge (UDMA-in only)	top level design
ti_dh	hold time ata_iordy edge to ata_data (UDMA-in only)	top level design
tco	propagation delay bus clock L-to-H to ata_cs0, ata_cs1, ata_da2, ata_da1, ata_da0, ata_dior, ata_diow, ata_dmack, ata_data, ata_buffer_en	top level design
tsu	set-up time ata_data to bus clock L-to-H	top level design
tsui	set-up time ata_iordy to bus clock H-to-L	top level design

Table 46-2. Timing Parameters (continued)

Name	Meaning	Controlled by
thi	hold time ata_iordy to bus clock H to L	top level design
tskew1	Max difference in propagation delay bus clock L-to-H to any of following signals ata_cs0, ata_cs1, ata_da2, ata_da1, ata_da0, ata_dior, ata_diow, ata_dmack, ata_data (write), ata_buffer_en	top level design
tskew2	Max difference in buffer propagation delay for any of following signals ata_cs0, ata_cs1, ata_da2, ata_da1, ata_da0, ata_dior, ata_diow, ata_dmack, ata_data (write), ata_buffer_en	transceiver
tskew3	Max difference in buffer propagation delay for any of following signals ata_iordy, ata_data (read)	transceiver
tbuf	Max buffer propagation delay	transceiver
tcable1	cable propagation delay for ata_data	cable
tcable2	cable propagation delay for control signals ata_dior, ata_diow, ata_iordy, ata_dmack	cable
tskew4	Max difference in cable propagation delay between ata_iordy and ata_data (read)	cable
tskew5	Max difference in cable propagation delay between (ata_dior, ata_diow, ata_dmack) and ata_cs0, ata_cs1, ata_da2, ata_da1, ata_da0, ata_data(write)	cable
tskew6	Max difference in cable propagation delay without accounting for ground bounce	cable

46.2.3.2 PIO Mode Timing

A timing diagram for PIO read mode is given in Figure 46-2.

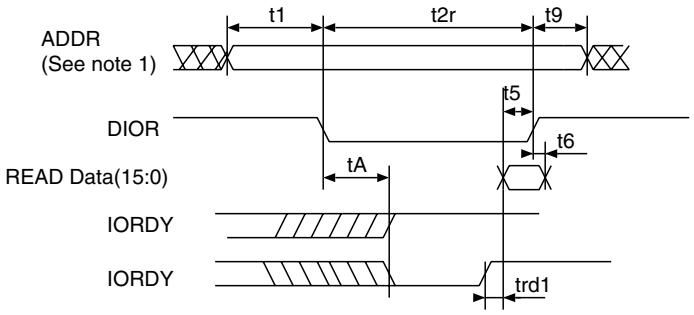


Figure 46-2. PIO Read Mode Timing

To fulfill read mode timing, the different timing parameters given in Table 46-3 must be observed.

Table 46-3. Timing Parameters PIO Read

ATA Parameter	PIO Read Mode Timing Parameter[1]	Value	How to meet?
t1	t1	t1(min) = time_1 × T - (tskew1 + tskew2 + tskew5)	time_1
t2	t2r	t2(min) = time_2r × T - (tskew1 + tskew2 + tskew5)	time_2r
t9	t9	t9(min) = time_9 × T - (tskew1 + tskew2 + tskew6)	time_9

Table 46-3. Timing Parameters PIO Read (continued)

ATA Parameter	PIO Read Mode Timing Parameter[1]	Value	How to meet?
t5	t5	t5(min) = tco + tsu + tbuf + tbuf + tcable1 + tcable2	if not met, increase time_2
t6	t6	0	
tA	tA	tA(min) = (1.5 + time_ax) × T - (tco + tsui + tcable2 + tcable2 + 2 × tbuf)	time_ax
trd	trd1	trd1(max) = (–trd) + (tskew3 + tskew4) trd1(min) = (time_pio_rdx - 0.5) × T – (tsu + thi) (time_pio_rdx - 0.5) × T > tsu + thi + tskew3 + tskew4	time_pio_rdx
t0	—	t0(min) = (time_1 + time_2 + time_9) × T	time_1, time_2r, time_9

1 See Figure 46-2.

In PIO write mode, timing waveforms are somewhat different. A timing diagram is given in Figure 46-3.

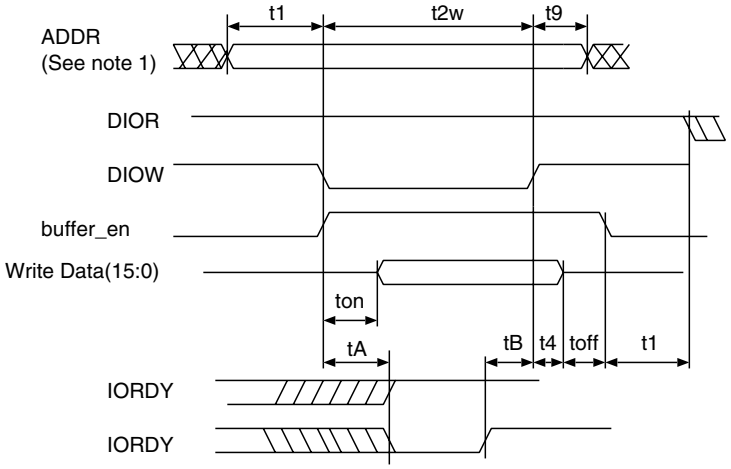


Figure 46-3. PIO Write Mode Timing

To fulfill this timing, several parameters need to be observed. See Table 46-4 for details on PIO write mode timing.

Table 46-4. Timing Parameters PIO Write

ATA Parameter	PIO Write Mode Timing Parameter[1]	Value	How to Meet?
t1	t1	t1(min) = time_1 × T – (tskew1 + tskew2 + tskew5)	time_1
t2	t2w	t2(min) = time_2w × T – (tskew1 + tskew2 + tskew5)	time_2w
t9	t9	t9(min) = time_9 × T – (tskew1 + tskew2 + tskew6)	time_9

Table 46-4. Timing Parameters PIO Write (continued)

ATA Parameter	PIO Write Mode Timing Parameter[1]	Value	How to Meet?
t3	—	t3(min) = (time_2w – time_on) × T – (tskew1 + tskew2 + tskew5)	if not met, increase time_2w
t4	t4	t4(min) = time_4 × T – tskew1	time_4
tA	tA	tA = (1.5 + time_ax) × T – (tco + tsui + tcable2 + tcable2 + 2 × tbuf)	time_ax
t0	—	t0(min) = (time_1 + time_2 + time_9) × T	time_1, time_2r, time_9
—	—	Avoid bus contention when switching buffer on by making ton long enough	—
—	—	Avoid bus contention when switching buffer off by making toff long enough	—

1 See Figure 46-3.

46.2.3.3 Timing in Multiword DMA Mode

In multi-word DMA mode, see Figure 46-4 for read timing diagram and Figure 46-5 for write timing diagram.

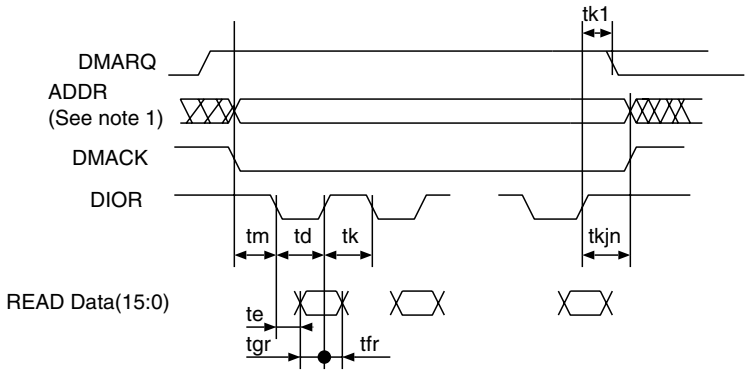


Figure 46-4. MDMA Read Timing

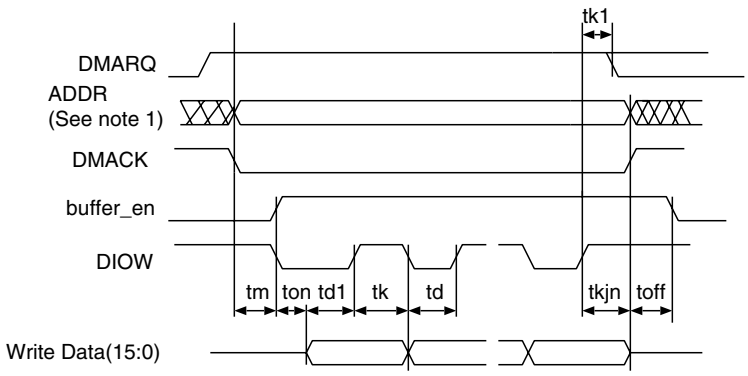


Figure 46-5. MDMA Write Timing

To meet the timing requirement, a number of timing parameters must be controlled. See Table 46-5 for details on timing parameters for MDMA read and write.

Table 46-5. Timing Parameters MDMA Read and Write

ATA Parameter	MDMA Read TIming[1] and MDMA Write TIming[2]	Value	How to Meet?
tm, ti	tm	tm(min) = ti(min) = time_m × T – (tskew1 + tskew2 + tskew5)	time_m
td	td, td1	td1(min) = td(min) = time_d × T – (tskew1 + tskew2 + tskew6)	time_d
tk	tk	tk(min) = time_k × T – (tskew1 + tskew2 + tskew6)	time_k
t0	-	t0(min) = (time_d + time_k) × T	time_d, time_k
tg(read)	tgr	tgr(min–read) = tco + tsu + tbuf + tbuf + tcable1 + tcable2 tgr(min–drive) = td – te(drive)	time_d
tf(read)	tfr	tfr(min–drive) = 0k	—
tg(write)	—	tg(min–write) = time_d × T –(tskew1 + tskew2 + tskew5)	time_d
tf(write)	—	tf(min–write) = time_k × T – (tskew1 + tskew2 + tskew6)	time_k
tL	—	tL(max) = (time_d + time_k–2)×T – (tsu + tco + 2×tbuf + 2×tcable2)	time_d, time_k[3]
tn, tj	tkjn	tn= tj= tkjn = (max(time_k,. time_jn) × T – (tskew1 + tskew2 + tskew6)	time_jn
—	ton toff	ton = time_on × T – tskew1 toff = time_off × T – tskew1	—

1 See Figure 46-4.

2 See Figure 46-5.

3 tk1 in the UDMA figures equals (tk -2*T)

46.2.3.4 UDMA In Timing Diagrams

UDMA mode timing is more complicated than PIO mode or MDMA mode. In this section, timing diagrams for UDMA in are provided.

Figure 46-6 shows timing for UDMA in transfer start.

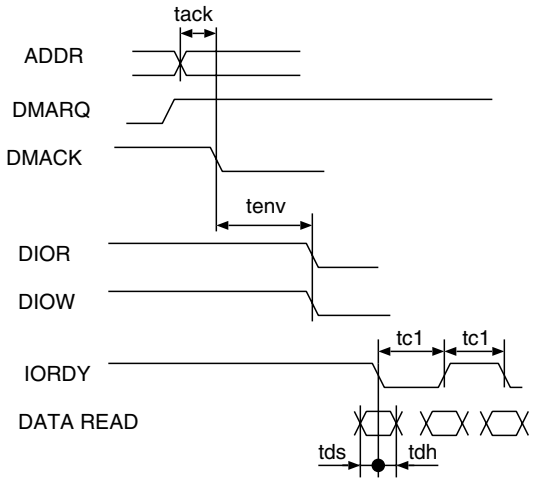


Figure 46-6. UDMA in Transfer Start Timing Diagram

Figure 46-7 shows timing for host terminating UDMA in transfer.

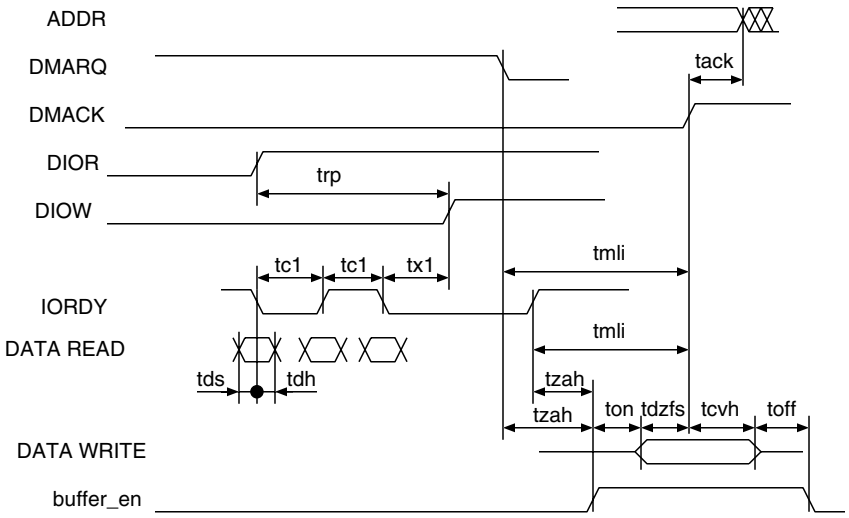


Figure 46-7. UDMA in Host Terminates Transfer

Figure 46-8 shows timing for device terminating UDMA in transfer.

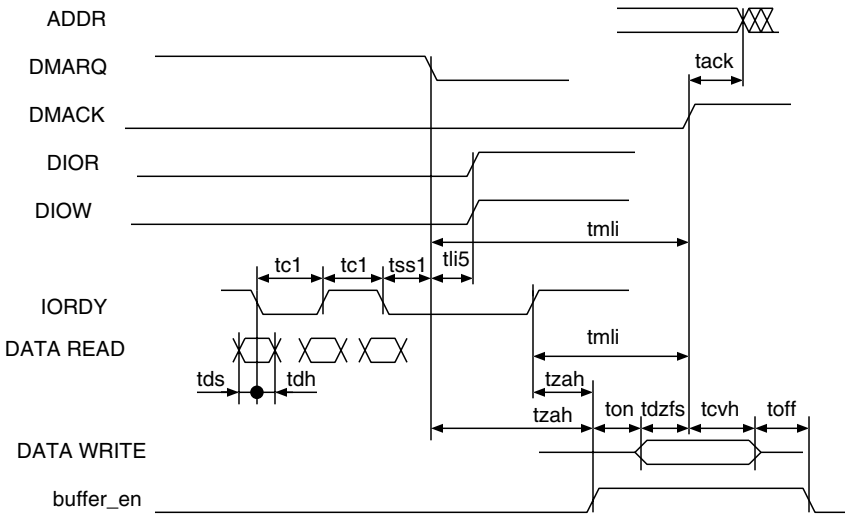


Figure 46-8. UDMA in Device Terminates Transfer

Timing parameters for UDMA in burst are listed in Table 46-6.

Table 46-6. Timing Parameters for UDMA in Burst

ATA Parameter	Spec Parameter	Value	Required Conditions
tack	tack	tack(min) = (time_ack × T) – (tskew1 + tskew2)	time_ack
tenv	tenv	tenv(min) = (time_env × T) – (tskew1 + tskew2) tenv(max) = (time_env × T) + (tskew1 + tskew2)	time_env
tds	tds1	tds – (tskew3) – ti_ds > 0	tskew3, ti_ds, ti_dh should be low enough
tdh	tdh1	tdh – (tskew3) –ti_dh > 0	
tcyc	tc1	(tcyc – tskew + **TBD**) > T	T big enough
trp	trp	trp(min) = time_rp × T – (tskew1 + tskew2 + tskew6)	time_rp
	tx1[1]	(time_rp × T) – (tco + tsu + 3T + 2 ×tbuf + 2×tcable2) > trfs (drive)	time_rp
tmli	tmli1	tmli1(min) = (time_mlix + 0.4) × T	time_mlix
tzah	tzah	tzah(min) = (time_zah + 0.4) × T	time_zah
tdzfs	tdzfs	tdzfs = (time_dzfs × T) – (tskew1 + tskew2)	time_dzfs
tcvh	tcvh	tcvh = (time_cvh ×T) – (tskew1 + tskew2)	time_cvh
—	ton toff	ton = time_on × T – tskew1 toff = time_off × T – tskew1	—

1 There is a special timing requirement in the ATA host that requires the internal DIOW to go only high three clocks after the last active edge on the DSTROBE signal. The equation given on this line tries to capture this constraint.

2. Make ton and toff big enough to avoid bus contention.

46.2.3.5 UDMA Out Timing Diagrams

UDMA mode timing is more complicated than PIO mode or MDMA mode. This section provides timing diagrams for UDMA out.

Figure 46-9 shows timing for UDMA out transfer start.

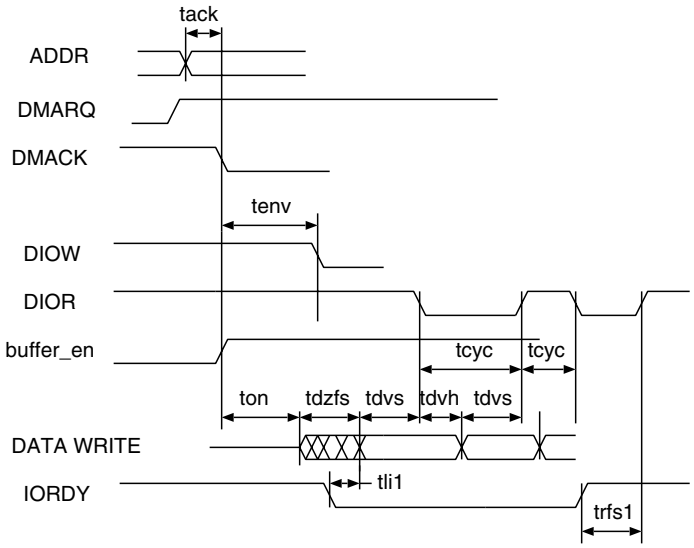


Figure 46-9. UDMA Out Transfer Start Timing Diagram

Figure 46-10 shows timing for host terminating UDMA out transfer.

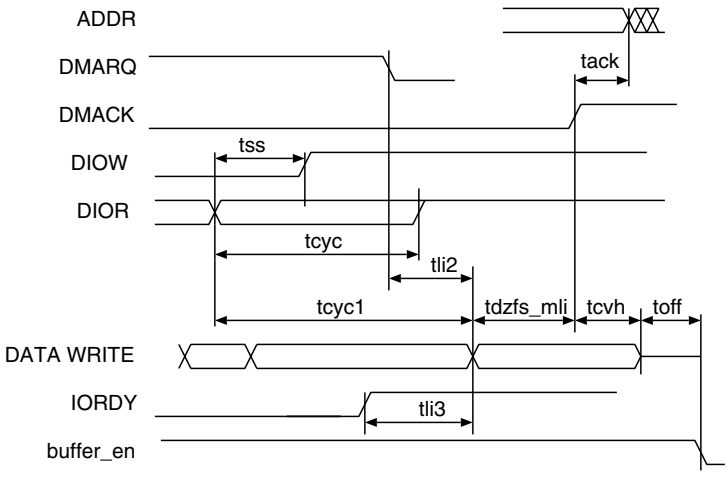


Figure 46-10. UDMA Out Host Terminates Transfer

Figure 46-11 shows the timing for device terminating UDMA out transfer.

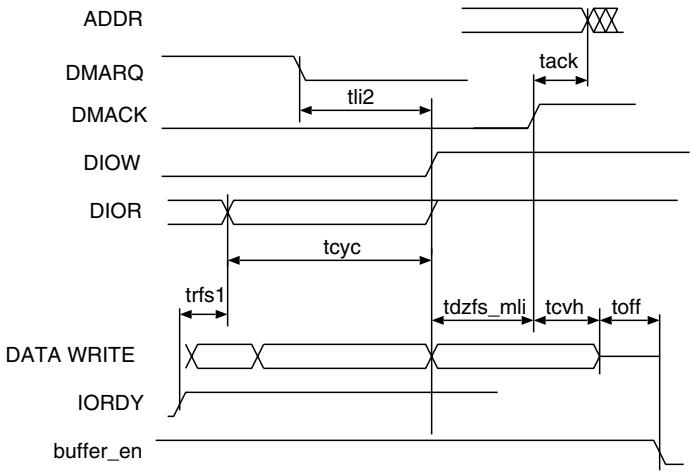


Figure 46-11. UDMA Out Device Terminates Transfer

Timing parameters for UDMA out burst are listed in Table 46-7.

Table 46-7. Timing Parameters UDMA Out Burst

ATA Parameter	Spec Parameter	Value	How to meet ?
tack	tack	tack(min) = (time_ack × T) – (tskew1 + tskew2)	time_ack
tenv	tenv	tenv(min) = (time_env × T) – (tskew1 + tskew2) tenv(max) = (time_env × T) + (tskew1 + tskew2)	time_env
tdvs	tdvs	tdvs = (time_dvs × T) – (tskew1 + tskew2)	time_dvs
tdvh	tdvh	tdvs = (time_dvh × T) – (tskew1 + tskew2)	time_dvh
tcyc	tcyc	tcyc = time_cyc × T – (tskew1 + tskew2)	time_cyc
t2cyc		t2cyc = time_cyc × 2 × T	time_cyc
trfs1	trfs	trfs = 1.6 × T + tsui + tco + tbuf + tbuf	—
—	tdzfs	tdzfs = time_dzfs × T – (tskew1)	time_dzfs
tss	tss	tss = time_ss × T – (tskew1 + tskew2)	time_ss
tmli	tdzfs_mli	tdzfs_mli =max(time_dzfs, time_mli) × T – (tskew1 + tskew2)	—
tli	tli1	tli1 > 0	—
tli	tli2	tli2 > 0	—
tli	tli3	tli3 > 0	—
tcvh	tcvh	tcvh = (time_cvh ×T) – (tskew1 + tskew2)	time_cvh
—	ton toff	ton = time_on × T – tskew1 toff = time_off × T – tskew1	—

46.3 Memory Map and Register Definition

Section 46.3.3, Register Descriptions," provides the detailed descriptions for all ATA registers.

46.3.1 Memory Map

Table 46-8 shows the ATA memory map.

Table 46-8. ATA Memory Map

Address	Register	Description	Access	Reset Value	Section/Page
0xBASE+0x000 (TIME_OFF)	TIME_OFF	transceiver timing parameter. Controls toff	R/W	0x01	46.3.3.2.1/46-21
0xBASE+0x001 (TIME_ON)	TIME_ON	transceiver timing parameter. Controls ton	R/W	0x01	46.3.3.2.2/46-21
0xBASE+0x002 (TIME_1)	TIME_1	PIO timing parameter. Controls t1	R/W	0x01	46.3.3.2.3/46-22
0xBASE+0x003 (TIME_2W)	TIME_2W	PIO timing parameter. Controls t2 during write cycles	R/W	0x01	46.3.3.2.4/46-22
0xBASE+0x004 (TIME_2R)	TIME_2R	PIO timing parameter. Controls t2 during read cycles	R/W	0x01	46.3.3.2.5/46-22
0xBASE+0x005 (TIME_AX)	TIME_AX	PIO timing parameter. Controls tA	R/W	0x01	46.3.3.2.6/46-23
0xBASE+0x00F (TIME_PIO_RDX)	TIME_PIO_RDX	PIO timing parameter. Controls trd	R/W	0x01	46.3.3.2.7/46-23
0xBASE+0x007 (TIME_4)	TIME_4	PIO timing parameter. Controls t4	R/W	0x01	46.3.3.2.8/46-23
0xBASE+0x008 (TIME_9)	TIME_9	PIO timing parameter. Controls t9	R/W	0x01	46.3.3.2.9/46-24
0xBASE+0x009 (TIME_M)	TIME_M	MDMA timing parameter. Controls tm	R/W	0x01	46.3.3.2.10/46-24
0xBASE+0x00A (TIME_JN)	TIME_JN	MDMA timing parameter. Controls tn and tj	R/W	0x01	46.3.3.2.11/46-24
0xBASE+0x00B (TIME_D)	TIME_D	MDMA timing parameter. Controls td	R/W	0x01	46.3.3.2.12/46-25
0xBASE+0x00C (TIME_K)	TIME_K	MDMA timing parameter. Controls tk	R/W	0x01	46.3.3.2.13/46-25
0xBASE+0x00D (TIME_ACK)	TIME_ACK	UDMA timing parameter. Controls tack	R/W	0x01	46.3.3.2.14/46-25
0xBASE+0x00E (TIME_ENV)	TIME_ENV	UDMA timing parameter. Controls tenv	R/W	0x01	46.3.3.2.15/46-26
0xBASE+0x00F (TIME_PIO_RDX)	TIME_RPX	UDMA timing parameter. Controls trp	R/W	0x01	46.3.3.2.16/46-26
0xBASE+0x010 (TIME_ZAH)	TIME_ZAH	UDMA timing parameter. Controls tzah	R/W	0x01	46.3.3.2.17/46-26
0xBASE+0x011 (TIME_MLIX)	TIME_MLIX	UDMA timing parameter. Controls tmli	R/W	0x01	46.3.3.2.18/46-27
0xBASE+0x012 (TIME_DVH)	TIME_DVH	UDMA timing parameter. Controls tdvh	R/W	0x01	46.3.3.2.19/46-27

Table 46-8. ATA Memory Map (continued)

Address	Register	Description	Access	Reset Value	Section/Page
0xBASE+0x013 (TIME_DZFS)	TIME_DZFS	UDMA timing parameter. Controls tdzfs	R/W	0x01	46.3.3.2.20/46-27
0xBASE+0x014 (TIME_DVS)	TIME_DVS	UDMA timing parameter. Controls tdvs	R/W	0x01	46.3.3.2.21/46-28
0xBASE+0x015 (TIME_CVH)	TIME_CVH	UDMA timing parameter. Controls tcvh	R/W	0x01	46.3.3.2.22/46-28
0xBASE+0x016 (TIME_SS)	TIME_SS	UDMA timing parameter. Controls tss	R/W	0x01	46.3.3.2.23/46-28
0xBASE+0x017 (TIME_CYC)	TIME_CYC	UDMA timing parameter. Controls tcyc and t2cyc	R/W	0x01	46.3.3.2.24/46-29
0xBASE+0x01C (FIFO_DATA_16)	FIFO_DATA_16	16-bit wide or 32-bit wide data port to/from FIFO	R/W	0x————	46.3.3.3.1/46-29
0xBASE+0x018 (FIFO_DATA_32)	FIFO_DATA_32		R/W	0x———— ————	46.3.3.3.2/46-30
0xBASE+0x020 (FIFO_FILL)	FIFO_FILL	FIFO filling in halfwords	Read-only	0x00	46.3.3.3.3/46-30
0xBASE+0x024 (ATA_CONTROL)	ATA_CONTROL	ATA interface control register	R/W	0x00	46.3.3.5.1/46-32
0xBASE+0x028 (INTERRUPT_PENDING)	INTERRUPT_ PENDING	Interrupt pending register	Read-only	0x1 —	46.3.3.5.1/46-32
0xBASE+0x02C (INTERRUPT_ENABLE)	INTERRUPT_ ENABLE	Interrupt enable register	R/W	0x0 —	46.3.3.5.2/46-33
0xBASE+0x030 (INTERRUPT_CLEAR)	INTERRUPT_ CLEAR	Interrupt clear register	Write-only	0x— —	46.3.3.5.3/46-34
0xBASE+0x034 (FIFO_ALARM)	FIFO_ALARM	FIFO alarm threshold	R/W	0x00	46.3.3.7/46-35
0xBASE+0xA0 (DRIVE_DATA)	DRIVE_DATA	drive data register	16-bit RW	—	46.3.3.7/46-35
0xBASE+0xA4 (DRIVE_FEATURES)	DRIVE_ FEATURES	drive features register	R/W	—	46.3.3.7/46-35
0xBASE+0xA8 (DRIVE_SECTOR_COUNT)	DRIVE_SECTOR_ COUNT	drive sector count register	R/W	—	46.3.3.7/46-35
0xBASE+0xAC (DRIVE_SECTOR_NUM)	DRIVE_SECTOR_ NUM	drive sector number register	R/W	—	46.3.3.7/46-35
0xBASE+0xB0 (DRIVE_CYL_LOW)	DRIVE_CYL_ LOW	drive cylinder low register	R/W	—	46.3.3.7/46-35
0xBASE+0xB4 (DRIVE_CYL_HIGH)	DRIVE_CYL_ HIGH	drive cylinder high register	R/W	—	46.3.3.7/46-35
0xBASE+0xB8 (DRIVE_DEV_HEAD)	DRIVE_DEV_ HEAD	drive device head register	R/W	—	46.3.3.7/46-35
0xBASE+0xBC (DRIVE_COMMAND)	DRIVE_ COMMAND	drive command register	Write-only	—	46.3.3.7/46-35
0xBASE+0xBC (DRIVE_STATUS)	DRIVE_STATUS	drive status register	Read-only	—	46.3.3.7/46-35
0xBASE+0xD8 (DRIVE_ALT_STATUS)	DRIVE_ALT_ STATUS	Drive alternate status register	Read-only	—	46.3.3.7/46-35

Table 46-8. ATA Memory Map (continued)

Address	Register	Description	Access	Reset Value	Section/Page
0xBASE+0xD8 (DRIVE_CONTROL)	DRIVE_ CONTROL	Drive control register	Write-only	—	46.3.3.7/46-35

46.3.2 Register Summary

Figure 46-12 shows the key to the register fields and Table 46-9 shows the register figure conventions.

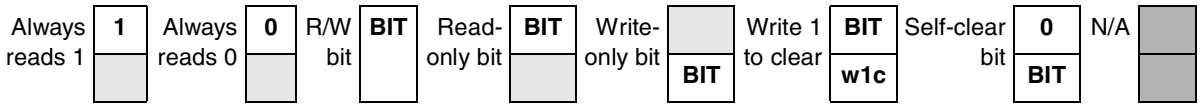


Figure 46-12. Key to Register Fields

Table 46-9. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 46-10 shows the ATA register summary.

Table 46-10. ATA Register Summary

Name		7	6	5	4	3	2	1	0
0xBASE+0x000 (TIME_OFF)	R	TIME_OFF[7:0]							
	W								

Table 46-10. ATA Register Summary (continued)

Name		7	6	5	4	3	2	1	0
0xBASE+0x001 (TIME_ON)	R	TIME_ON[7:0]							
	W								
0xBASE+0x002 (TIME_1)	R	TIME_1[7:0]							
	W								
0xBASE+0x003 (TIME_2W)	R	TIME_2W[7:0]							
	W								
0xBASE+0x004 (TIME_2R)	R	TIME_2R[7:0]							
	W								
0xBASE+0x005 (TIME_AX)	R	TIME_AX[7:0]							
	W								
0xBASE+0x00F (TIME_PIO_RDX)	R	TIME_RDX[7:0]							
	W								
0xBASE+0x007 (TIME_4)	R	TIME_4[7:0]							
	W								
0xBASE+0x008 (TIME_9)	R	TIME_9[7:0]							
	W								
0xBASE+0x009 (TIME_M)	R	TIME_M[7:0]							
	W								
0xBASE+0x00A (TIME_JN)	R	TIME_JN[7:0]							
	W								
0xBASE+0x00B (TIME_D)	R	TIME_D[7:0]							
	W								
0xBASE+0x00C (TIME_K)	R	TIME_K[7:0]							
	W								
0xBASE+0x00D (TIME_ACK)	R	TIME_ACK[7:0]							
	W								
0xBASE+0x00E (TIME_ENV)	R	TIME_ENV[7:0]							
	W								
0xBASE+0x00F (TIME_RPX)	R	TIME_RPX[7:0]							
	W								
0xBASE+0x010 (TIME_ZAH)	R	TIME_ZAH[7:0]							
	W								

Table 46-10. ATA Register Summary (continued)

Name		7	6	5	4	3	2	1	0
0xBASE+0x011 (TIME_MLIX)	R	TIME_MLIX[7:0]							
	W								
0xBASE+0x012 (TIME_DVH)	R	TIME_DVH[7:0]							
	W								
0xBASE+0x013 (TIME_DZFS)	R	TIME_DZFS[7:0]							
	W								
0xBASE+0x014 (TIME_DVS)	R	TIME_DVS[7:0]							
	W								
0xBASE+0x015 (TIME_CVH)	R	TIME_CVH[7:0]							
	W								
0xBASE+0x016 (TIME_SS)	R	TIME_SS[7:0]							
	W								
0xBASE+0x017 (TIME_CYC)	R	TIME_CYC[7:0]fifo_data[15:0]							
	W								

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x01C (FIFO_DATA_16)	R	fifo_data[15:0]															
	W																

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x018 (FIFO_DATA_32)	R	fifo_data[23:16]															
	W																
	R	fifo_data[15:0]															
	W																

Name		7	6	5	4	3	2	1	0
0xBASE+0x020 (FIFO_FILL)	R	FIGO_FILL[7:0]							
	W								
0xBASE+0x024 (ATA_CONTROL)	R	fifo_rst_b	ata_rst_b	fifo_tx_en	fifo_rcv_en	dma_pending	dma_ultra_selected	dma_write	iordy_en
	W								

Table 46-10. ATA Register Summary (continued)

Name		7	6	5	4	3	2	1	0
0xBASE+0x028 (INTERRUPT_PENDING)	R	ata_ intrq1	fifo_ under-flow	fifo_ over-flow	controll er_ idle	ata_ irtrq2			
	W								
0xBASE+0x02C (INTERRUPT_ENABLE)	R	ata_ intrq1	fifo_ under-flow	fifo_ over-flow	controll er_ idle	ata_ irtrq2			
	W								
0xBASE+0x030 (INTERRUPT_CLEAR)	R								
	W		fifo_ underflo w	fifo_ overflo w					
0xBASE+0x034 (FIFO_ALARM)	R	FIFO_ALARM[7:0]							
	W								

46.3.3 Register Descriptions

This section contains the detailed register descriptions for the ATA registers.

46.3.3.1 Endianness

The ATA interface works both in little-endian or big-endian mode. The addresses of all registers remain the same regardless of endianess mode. The few 16-bit and 32-bit registers represent strings of 2 or 4 bytes. The byte order in the 16-bit or 32-bit register is dependent on endianess selection.

- Little endian, 16-bit or 32-bit register:
  - bits [7:0]: byte 0
  - bits [15:8]: byte 1
  - bits [23:8]: byte 2
  - bits [31:24]: byte 3
- Big endian, 32-bit register
  - bits [31:24]: byte 0
  - bits [23:16]: byte 1
  - bits [15:8]: byte 2
  - bits [7:0]: byte 3
- Big endian, 16-bit register
  - bits [15:8]: byte 0
  - bits [7:0]: byte 1

46.3.3.2 Timing Registers

Registers (ata_base +$0) until (ata_base + $17) contain timing parameters. These timing parameters control the timing on the ATA bus as shown in details in the following illustrations:

- Section 46.2.3.2, PIO Mode Timing”
- Section 46.2.3.3, Timing in Multiword DMA Mode”
- Section 46.2.3.4, UDMA In Timing Diagrams”
- Section 46.2.3.5, UDMA Out Timing Diagrams”

Every timing parameter is 8-bit wide and can assume valid values between 1 and 255. Reset value is always 1.

All figures in this section show timing registers.

46.3.3.2.1 TIME_OFF Register

See Figure 46-13 for illustration of valid bits in the TIME_OFF Register and Table 46-8 for description of the bit fields.

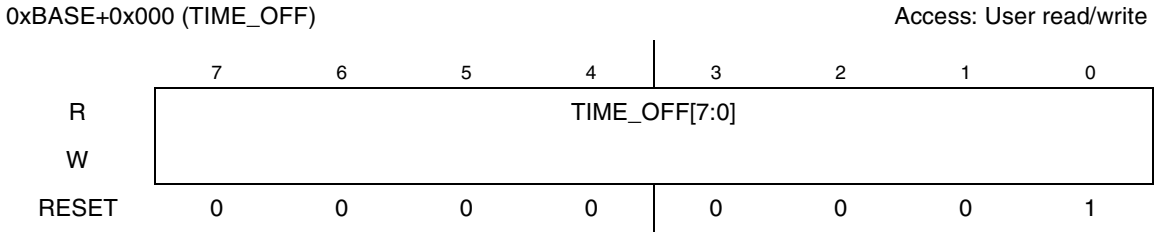


Figure 46-13. TIME_OFF Register

46.3.3.2.2 TIME_ON Register

See Figure 46-14 for illustration of valid bits in the TIME_ON Register and Table 46-8 for description of the bit fields.

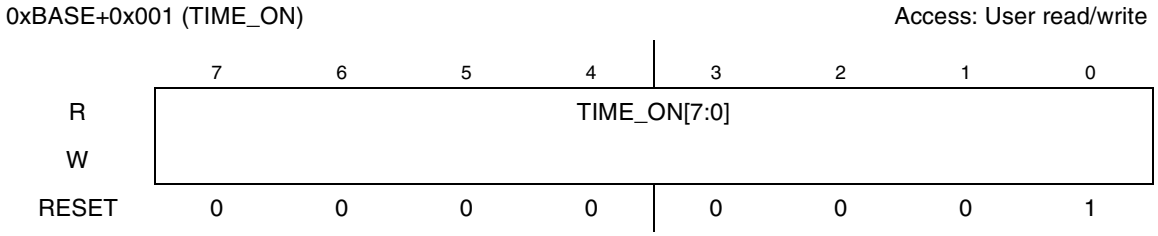


Figure 46-14. TIME_ON Register

46.3.3.2.3 TIME_1 Register

See Figure for illustration of valid bits in the TIME_1 Register and Table 46-8 for description of the bit fields.

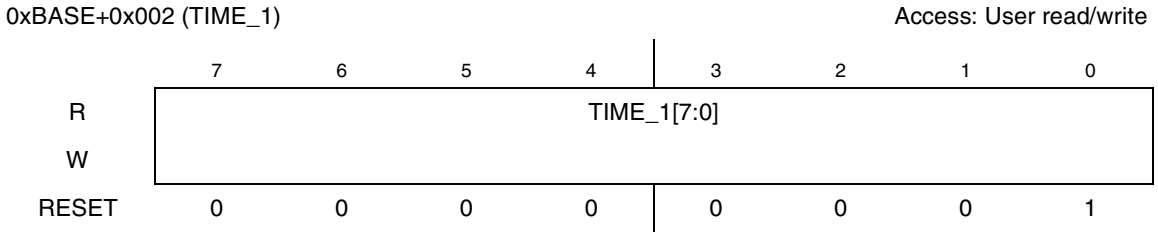


Figure 46-15. TIME_1 Register

46.3.3.2.4 TIME_2W Register

See Figure 46-16 for illustration of valid bits in the TIME_2W Register and Table 46-8 for description of the bit fields.

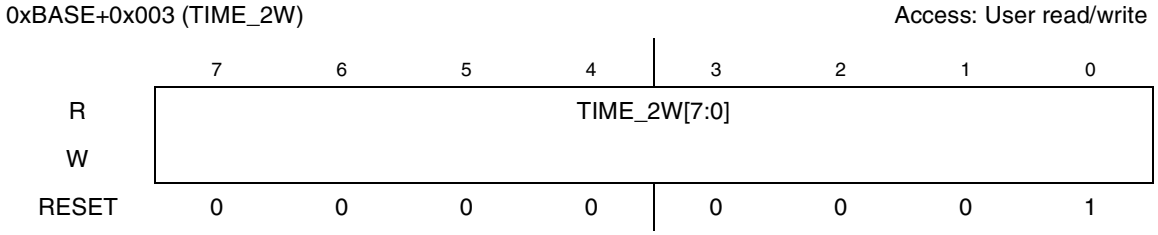


Figure 46-16. TIME_2W Register

46.3.3.2.5 TIME_2R Register

See Figure 46-17 for illustration of valid bits in the TIME_2R Register and Table 46-8 for description of the bit fields.

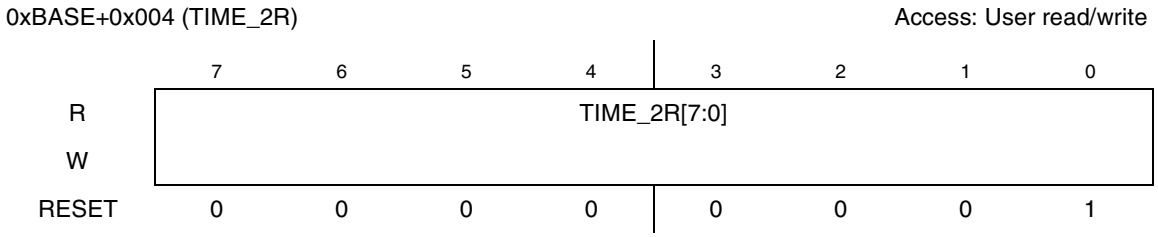


Figure 46-17. TIME_2R Register

46.3.3.2.6 TIME_AX Register

See Figure 46-18 for illustration of valid bits in the TIME_AX Register and Table 46-8 for description of the bit fields.

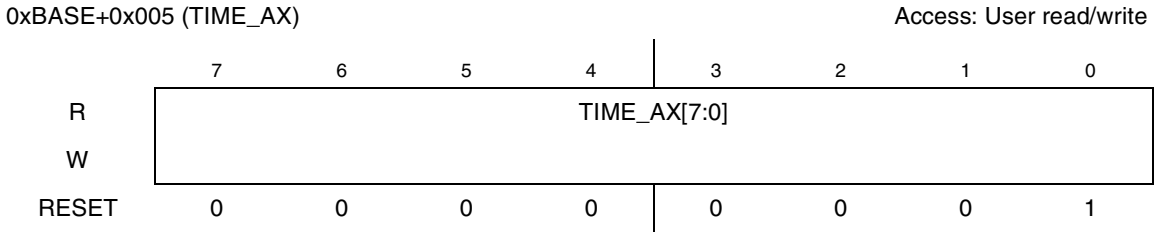


Figure 46-18. TIME_AX Register

46.3.3.2.7 TIME_PIO_RDX Register

See Figure 46-19 for illustration of valid bits in the TIME_PIO_RDX Register and Table 46-8 for description of the bit fields.

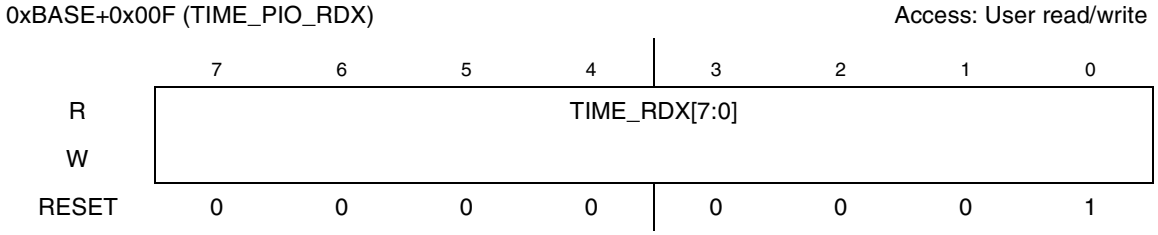


Figure 46-19. TIME_PIO_RDX Register

46.3.3.2.8 TIME_4 Register

See Figure 46-20 for illustration of valid bits in the TIME_4 Register and Table 46-8 for description of the bit fields.

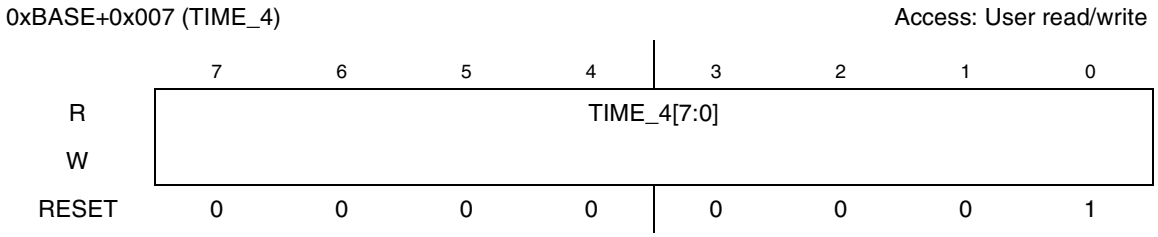


Figure 46-20. TIME_4 Register

46.3.3.2.9 TIME_9 Register

See Figure 46-21 for illustration of valid bits in the TIME_9 Register and Table 46-8 for description of the bit fields.

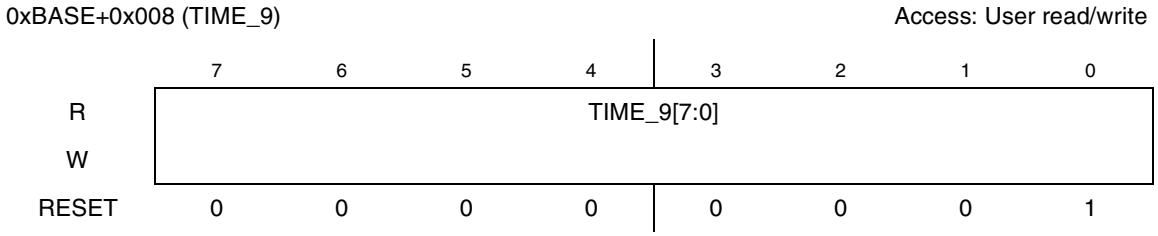


Figure 46-21. TIME_9 Register

46.3.3.2.10 TIME_M Register

See Figure 46-22 for illustration of valid bits in the TIME_M Register and Table 46-8 for description of the bit fields.

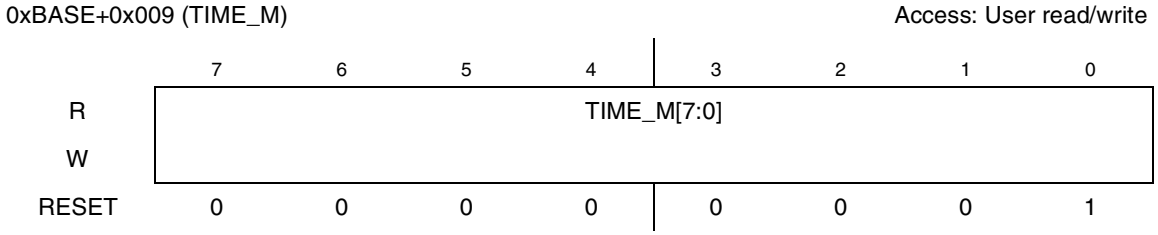


Figure 46-22. TIME_M

46.3.3.2.11 TIME_JN Register

See Figure 46-23 for illustration of valid bits in the TIME_JN Register and Table 46-8 for description of the bit fields.

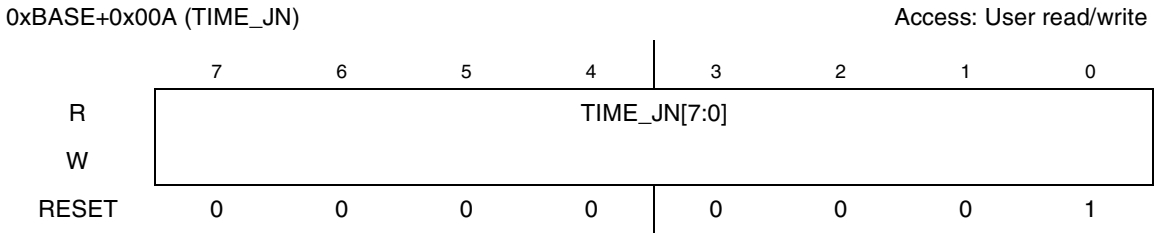


Figure 46-23. TIME_JN Register

46.3.3.2.12 TIME_D Register

See Figure 46-24 for illustration of valid bits in the TIME_D Register and Table 46-8 for description of the bit fields.

0xBASE+0x00B (TIME_D) Access: User read/write

	7	6	5	4	3	2	1	0
R	TIME_D[7:0]							
W								
RESET	0	0	0	0	0	0	0	1

Figure 46-24. TIME_D Register

46.3.3.2.13 TIME_K Register

See Figure 46-25 for illustration of valid bits in the TIME_K Register and Table 46-8 for description of the bit fields.

0xBASE+0x00C (TIME_K) Access: User read/write

	7	6	5	4	3	2	1	0
R	TIME_K[7:0]							
W								
RESET	0	0	0	0	0	0	0	1

Figure 46-25. TIME_K Register

46.3.3.2.14 TIME_ACK Register

See Figure 46-26 for illustration of valid bits in the TIME_ACK Register and Table 46-8 for description of the bit fields.

0xBASE+0x00D (TIME_ACK) Access: User read/write

	7	6	5	4	3	2	1	0
R	TIME_ACK[7:0]							
W								
RESET	0	0	0	0	0	0	0	1

Figure 46-26. TIME_ACK Register

46.3.3.2.15 TIME_ENV Register

See Figure 46-27 for illustration of valid bits in the TIME_ENV Register and Table 46-8 for description of the bit fields.

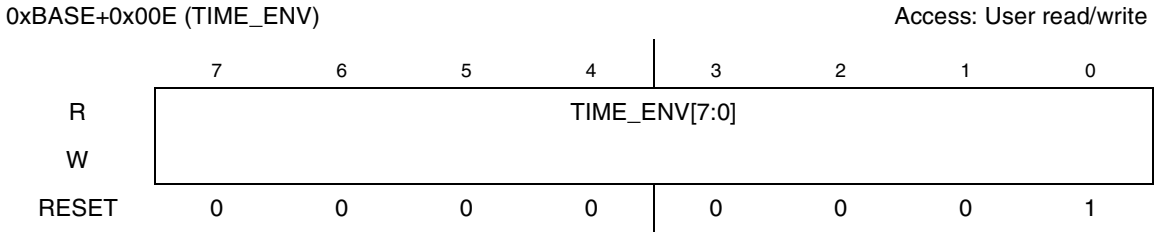


Figure 46-27. TIME_ENV Register

46.3.3.2.16 TIME_RPX Register

See Figure 46-28 for illustration of valid bits in the TIME_RPX Register and Table 46-8 for description of the bit fields.

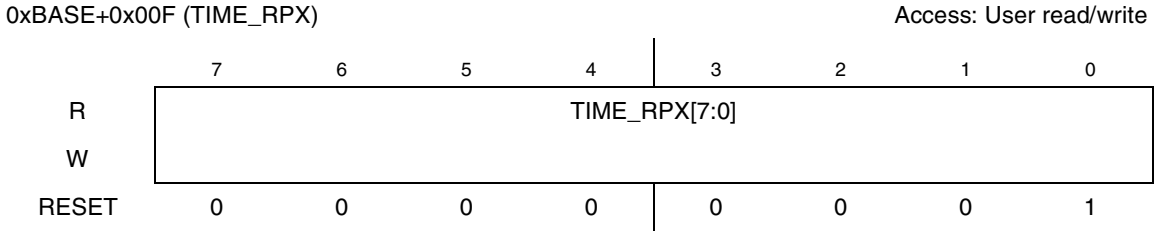


Figure 46-28. TIME_RPX Register

46.3.3.2.17 TIME_ZAH Register

See Figure 46-29 for illustration of valid bits in the TIME_ZAH Register and Table 46-8 for description of the bit fields.

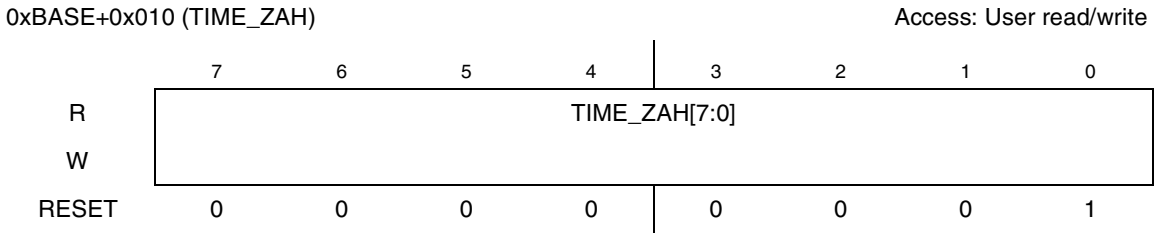


Figure 46-29. TIME_ZAH Register

46.3.3.2.18 TIME_MLIX Register

See Figure 46-30 for illustration of valid bits in the TIME_MLIX Register and Table 46-8 for description of the bit fields.

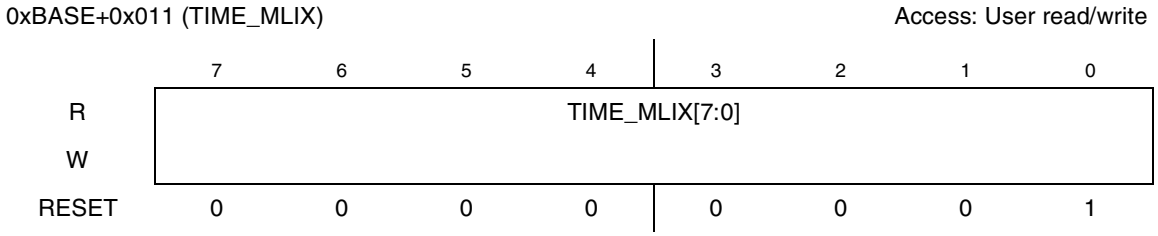


Figure 46-30. TIME_MLIX

46.3.3.2.19 TIME_DVH Register

See Figure 46-31 for illustration of valid bits in the TIME_DVH Register and Table 46-8 for description of the bit fields.

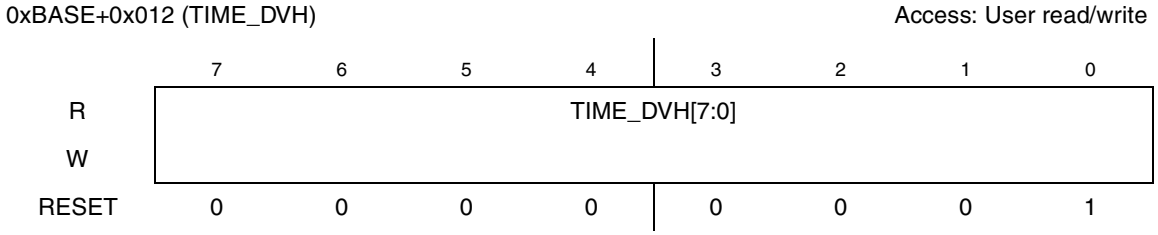


Figure 46-31. TIME_DVH Register

46.3.3.2.20 TIME_DZFS Register

See Figure 46-32 for illustration of valid bits in the TIME_DZFS Register and Table 46-8 for description of the bit fields.

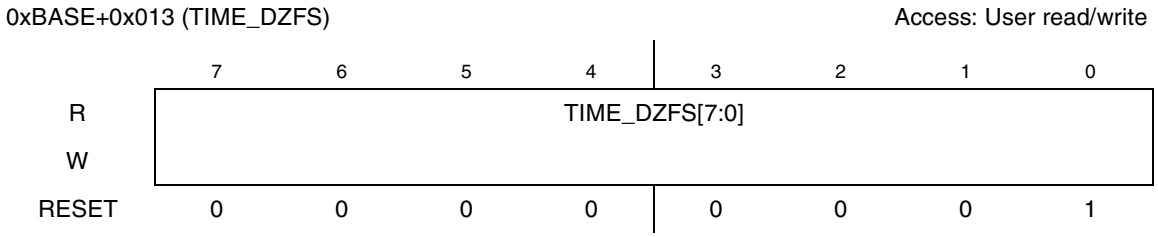


Figure 46-32. TIME_DZFS Register

46.3.3.2.21 TIME_DVS Register

See Figure 46-33 for illustration of valid bits in the TIME_DVS Register and Table 46-8 for description of the bit fields.

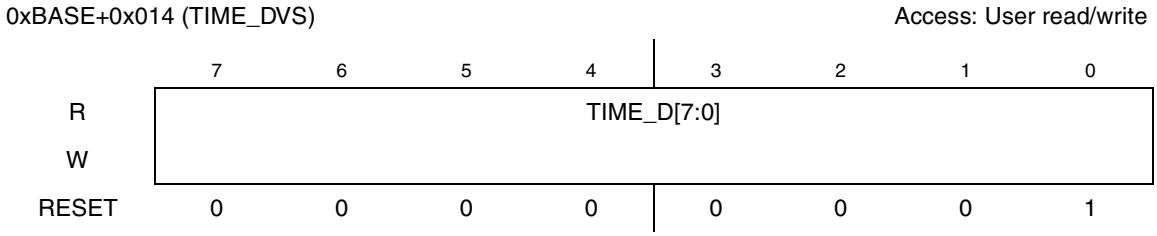


Figure 46-33. TIME_DVS

46.3.3.2.22 Time_CVH Register

See Figure 46-34 for illustration of valid bits in the TIME_CVH Register and Table 46-8 for description of the bit fields.

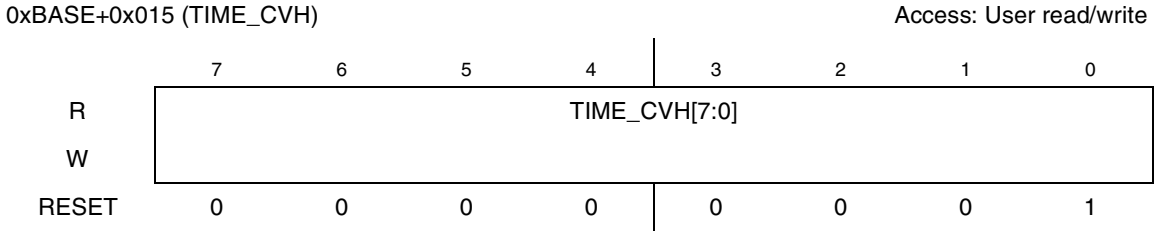


Figure 46-34. TIME_CVH Register

46.3.3.2.23 TIME_SS Register

See Figure 46-35 for illustration of valid bits in the TIME_SS Register and Table 46-8 for description of the bit fields.

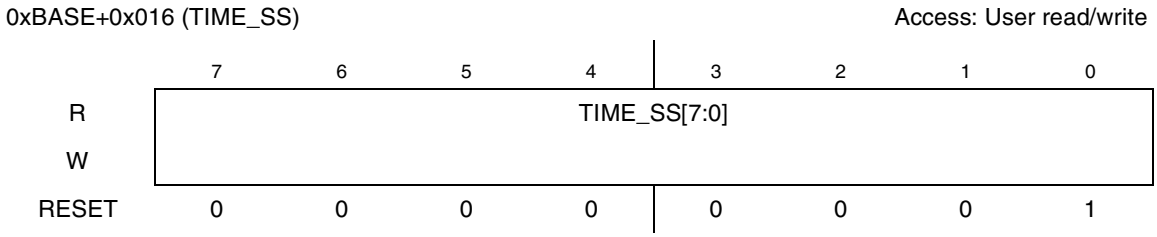


Figure 46-35. TIME_SS Register

46.3.3.2.24 TIME_CYC Register

See Figure 46-36 for illustration of valid bits in the TIME_CYC Register and Table 46-8 for description of the bit fields.

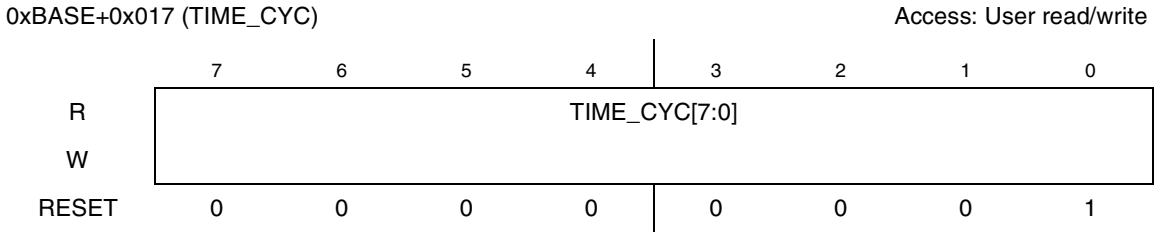


Figure 46-36. TIME_CYC

46.3.3.3 FIFO Data Registers

The following subsections provide figures for the FIFO Data register in 16- and 32-bit mode.

46.3.3.3.1 FIFO_Data Register in 16-bit Mode

See Figure 46-37 for illustration of valid bits in the FIFO_Data Register in 16-bit mode and Table 46-8 for description of the bit fields.

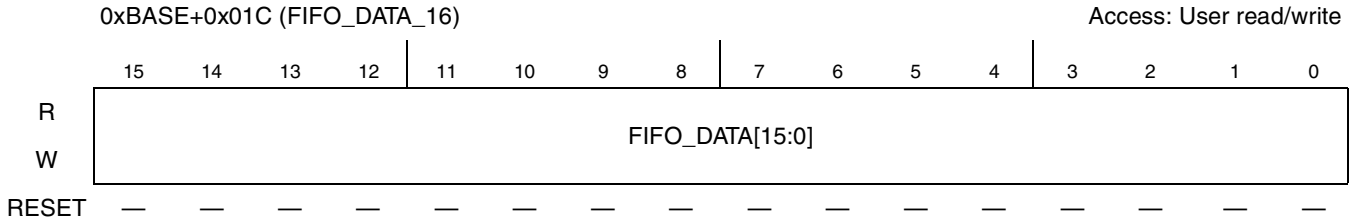


Figure 46-37. FIFO_Data Register In 16-bit Mode

46.3.3.3.2 FIFO_Data Register in 32-bit Mode

See Figure 46-38 for illustration of valid bits in the FIFO_Data Register in 32-bit mode and Table 46-8 for description of the bit fields.

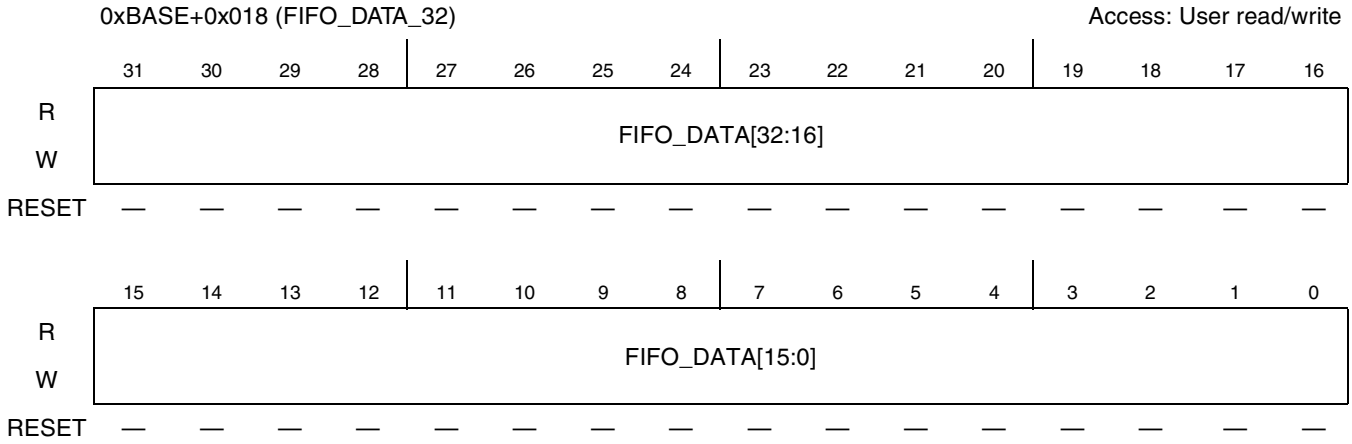


Figure 46-38. FIFO_Data Register in 32-bit Mode

The FiFO_DATA register is used to read or write data to the internal FIFO. It can be accessed as a 16-bit register or as a 32-bit register. Any long write to the register will put the four bytes written into the FIFO. Any word write will put the two bytes written into the FIFO. Any long read will read four bytes from the FIFO. Any word read will read two bytes from the FIFO.

46.3.3.3.3 FIFO_FILL Register

See Figure 46-39 for illustration of valid bits in the FIFO_FILL register and Table 46-8 for description of the bit fields.

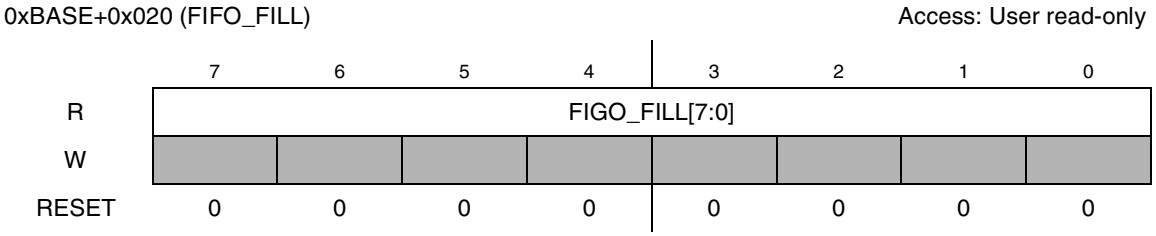


Figure 46-39. FIFO_FILL Register

FIFO_FILL is a read-only register. Any read to it returns the current number of halfwords present in the fifo.

46.3.3.4 ATA_CONTROL Register

See Figure 46-40 for illustration of valid bits in the ATA Control Register and Table 46-11 for description of the bit fields.

0xBASE+0x024 (ATA_CONTROL) Access: User read/write

	7	6	5	4	3	2	1	0
R W	fifo_rst_b	ata_rst_b	fifo_tx_en	fifo_rcv_en	dma_pending	dma_ultra_selected	dma_write	iordy_en
RESET	0	0	0	0	0	0	0	0

Figure 46-40. ATA Control Register

Table 46-11. ATA Control Register Field Descriptions

Field	Description
7 fifo_rst_b	This field controls if the internal FIFO is in reset or enabled 0 FIFO reset 1 FIFO normal operation
6 ata_rst_b	This bit controls the level on the ata_reset_b pin, and controls the reset of the internal ata protocol engine. 0 ata_reset_b = 0, ata drive is reset, and internal protocol engine reset. 1 ata_reset_b = 1, ata drive is not reset and internal protocol engine normal operation.
5 fifo_tx_en	FIFO transmit enable. This bit controls if the FIFO will make transmit data requests to the DMA. If enabled, the FIFO will request the DMA to refill it whenever FIFO filling drops below the alarm level. 0 FIFO refill by DMA disabled 1 FIFO refill by DMA enabled
4 fifo_rcv_en	FIFO receive enable. This bit controls if the FIFO will make receive data requests to the DMA. If enabled, the FIFO will request the DMA to empty it whenever FIFO filling becomes greater or equal to the alarm level. 0 FIFO empty by DMA disabled 1 FIFO empty by DMA enabled
3 dma_pending	DMA pending bit. This bit controls if the ATA interface will respond to a DMA request originating in the drive. If this bit is asserted, the ATA interface will start a multiword DMA or ultra DMA burst whenever the drive asserts ata_dmarq. 0 ATA interface will not start DMA burst 1 ATA interface will start multiword DMA or ultra DMA burst whenever drive asserts dmarq
2 dma_ultra_selected	This bit indicates if a DMA burst started, the UDMA or MDMA protocol will be used 0 Multiword DMA protocol will be used 1 Ultra DMA protocol will be used
1 dma_write	This bit indicates the data direction on any DMA burst started 0 DMA in burst, ATA interface reads from drive 1 DMA out burst, ATA interface writes to drive
0 iordy_en	This bit indicates if the ata_iordy handshake will be used during PIO mode 0 IORDY will be disregarded 1 IORDY handshake will be used

46.3.3.5 Interrupt Registers

A group of three registers control the interrupt interface from the ATA module and going to the CPU and DMA. There are two interrupts controlled by these registers:

- ipbus_int. This interrupt is controlled by bits 3,4, 5 and 6 of the interrupt registers. It will be asserted if one of the 4 bits is set in the interrupt_pending register, while the same bit is set in the interrupt_enable register. This interrupt goes to the CPU.
- fifo_txfer_end_alarm. This interrupt is controlled by bit 7 of the interrupt registers. If ata_intrq1 is set in both the interrupt enable and interrupt pending register, fifo_txfer_end_alarm will be asserted. The goal of this interrupt is to inform the DMA that the running data transfer has ended. This interrupt goes to the smart DMA.

These three registers have mostly the same bits. If a bit is set in the interrupt pending register, its interrupt is pending, and will produce an interrupt if the same bit is set in the interrupt enable register. Some bits in the interrupt pending register are sticky bits. Writing a ‘1’ to the corresponding bit in the interrupt clear bit, will reset them.

46.3.3.5.1 Interrupt_Pending Register

See Figure 46-41 for illustration of valid bits in the Interrupt_Pending Register and Table 46-12 for description of the bit fields.

0xBASE+0x028 (INTERRUPT_PENDING) Access: User read-only

	7	6	5	4	3	2	1	0
R	ata_intrq1	fifo_ underflow	fifo_ overflow	controller_ idle	ata_irtrq2			
W								
RESET	0[1]	0	0	1	0	—	—	—

Figure 46-41. Interrupt_Pending Register

[1] Interrupts ata_intrq1 and ata_intrq2 only reset to 0 if during reset the interrupt input is low.

Table 46-12. Interrupt Pending Register Field Description

Field	Description
7 ata_intrq1	ATA interrupt request 1 This bit reflects the value of the ata_intrq interrupt input. It is set in the interrupt pending register when the drive interrupt is pending, cleared otherwise. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, fifo_txfer_end_alarm will be asserted, signalling the DMA the end of the transfer. The interrupt clear register has no influence on this bit.
6 fifo_underflow	FIFO underfow This bit reports FIFO underflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO underflow condition. It is cleared by writing a ‘1’ to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.

Table 46-12. Interrupt Pending Register Field Description (continued)

Field	Description
5 fifo_overflow	FIFO overflow This bit reports FIFO overflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO overflow condition. It is cleared by writing a '1' to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.
4 controller_idle	Controller Idle This bit reports controller idle. It is set when the ATA protocol engine is idle, there is no activity on the ATA bus. It is cleared when there is activity on the ATA bus. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu. The interrupt clear register has no influence on this bit.
3 ata_intrq2	ATA interrupt request 2 This bit reflects the value of the ata_intrq interrupt input. It is set in the interrupt pending register when the drive interrupt is pending, cleared otherwise. It has exactly same functioning as ata_intrq1, but this bit affects ipbus_int, while the other affects interrupt to the DMA. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be asserted, signalling the CPU the drive is requesting attention. The interrupt clear register has no influence on this bit.
2–0 Uncommitted	N/A

46.3.3.5.2 Interrupt_Enable Register

See Figure 46-42 for illustration of valid bits in the Interrupt_Enable Register and Table 46-13 for description of the bit fields.

0xBASE+0x02C (INTERRUPT_ENABLE) Access: User read/write

	7	6	5	4	3	2	1	0
R W	ata_intrq1	fifo_ underflow	fifo_ overflow	controller_ idle	ata_irtrq2			
RESET	0	0	0	0	0	—	—	—

Figure 46-42. Interrupt_Enable Register

Table 46-13. Interrupt Enable Register Field Description

Field	Description
7 ata_intrq1	ATA interrupt request 1 This bit reflects the value of the ata_intrq interrupt input. It is set in the interrupt pending register when the drive interrupt is pending, cleared otherwise. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, fifo_txfer_end_alarm will be asserted, signalling the DMA the end of the transfer. The interrupt clear register has no influence on this bit.
6 fifo_underflow	FIFO underflow This bit reports FIFO underflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO underflow condition. It is cleared by writing a '1' to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.
5 fifo_overflow	FIFO overflow This bit reports FIFO overflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO overflow condition. It is cleared by writing a '1' to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.
4 controller_idle	Controller Idle This bit reports controller idle. It is set when the ATA protocol engine is idle, there is no activity on the ATA bus. It is cleared when there is activity on the ATA bus. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu. The interrupt clear register has no influence on this bit.
3 ata_intrq2	ATA interrupt request 2 This bit reflects the value of the ata_intrq interrupt input. It is set in the interrupt pending register when the drive interrupt is pending, cleared otherwise. It has exactly same functioning as ata_intrq1, but this bit affects ipbus_int, while the other affects interrupt to the DMA. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be asserted, signalling the CPU the drive is requesting attention. The interrupt clear register has no influence on this bit.
2–0 Uncommitted	N/A

46.3.3.5.3 Interrupt_Clear Register

See Figure 46-43 for illustration of valid bits in the Interrupt_Clear Register and Table 46-14 for description of the bit fields.

0xBASE+0x030 (INTERRUPT_CLEAR) Access: User write-only

	7	6	5	4	3	2	1	0
R								
W		fifo_ underflow	fifo_ overflow					
RESET	—	—	—	—	—	—	—	—

Figure 46-43. Interrupt_Clear Register

Table 46-14. Interrupt Clear Register Field Description

Field	Description
7 Uncommitted	N/A
6 fifo_underflow	FIFO underfow This bit reports FIFO underflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO underflow condition. It is cleared by writing a ‘1’ to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.
5 fifo_overflow	FIFO overflow This bit reports FIFO overflow. Sticky bit. It is set in the interrupt pending register when there is a FIFO overflow condition. It is cleared by writing a ‘1’ to this bit in the interrupt clear register. When the bit is set in the interrupt pending register, and the same bit is set in the interrupt enable register, ipbus_int will be active, signalling interrupt to the cpu.
4–0 Uncommitted	N/A

46.3.3.6 FIFO Alarm Register

See Figure 46-44 for illustration of valid bits in the FIFO_Alarm Register.

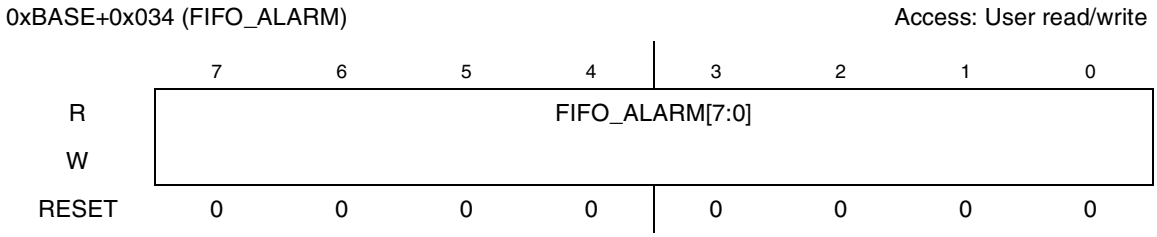


Figure 46-44. FIFO_Alarm Register

This register contains the threshold to generate fifo_rcv_alarm and fifo_tx_alarm to the DMA interface.

- If (`fifo_tx_enable == 1 && fifo_fill < fifo_alarm`): fifo_tx_alarm is set 1, request is made to DMA to refill fifo.
- If (`fifo_rcv_alarm == 1 && fifo_fill >= fifo_alarm`): fifo_rcv_alarm is set 1, request is made to DMA to empty fifo.

46.3.3.7 Drive Registers connected to ATA Bus

Some registers are addressable, but are not present in the ATA interface module. A list is given in Table 46-15. If a read or write access is made to one of these registers, the read or write is mapped to a PIO read or write cycle on the ATA bus, and the corresponding register in the device attached to the ATA bus is accessed.

If the drive_data register is accessed while the ATA interface operates in big-endian mode, the bytes to/from the ATA bus are swapped. No swaps occur in little-endian mode, nor for any other register.

Table 46-15. Drive Registers Connected to ATA Bus

Address	Name	Description	Access
0xBASE+0x0A0 (DRIVE_DATA)	drive_data	Drive data register	R/W
0xBASE+0x0A4 (DRIVE_FEATURES)	drive_features	Drive features register	R/W
0xBASE+0x0A8 (DRIVE_SECTOR_COUNT)	drive_sector_count	Drive sector count register	R/W
0xBASE+0x0AC (DRIVE_SECTOR_NUM)	drive_sector_num	Drive sector number register	R/W
0xBASE+0x0B0 (DRIVE_CYL_LOW)	drive_cyl_low	Drive cylinder low register	R/W
0xBASE+0x0B4 (DRIVE_CYL_HIGH)	drive_cyl_high	Drive cylinder high register	R/W
0xBASE+0x0B8 (DRIVE_DEV_HEAD)	drive_dev_head	Drive device head register	R/W
0xBASE+0x0BC (DRIVE_COMMAND)	drive_command	Drive command register	Write-only
0xBASE+0x0C0 (DRIVE_STATUS)	drive_status	Drive status register	Read-only
0xBASE+0x0C4 (DRIVE_ALT_STATUS)	drive_alt_status	Drive alternate status register	Read-only
0xBASE+0x0C8 (DRIVE_CONTROL)	drive_control	Drive control register	Write-only

46.4 Functional Description

The ATA interface provides two ways to communicate with the ATA peripherals connected to the ATA bus

- PIO mode read/write operation to the ATA bus.
- DMA transfers with the ATA bus

The operation of the peripheral is described in detail in the following sections.

46.4.1 Resetting ATA Bus

The ATA bus reset ata_reset_b is asserted whenever bit 6 ata_rst_b of register ata_control is cleared to 0. At the same time, the ATA protocol engine is reset. When this bit is set to 1, the reset is released.

46.4.2 Programming ATA Bus Timing and iordy_en

The timing the ATA interface will operate with on the ATA bus is programmable. The 24 timing registers at 0xBASE+0x0 to 0xBASE+0x17 are used for this. How these registers affect the timing parameters on the ATA bus is detailed in Section 46.2, External Signal Description.” It is allowed to reprogram these registers at any time when the ATA bus is idle, so before reprogramming make sure that:

- bit dma_pending in ata_control register is cleared.
- bit controller_idle in interrupt_pending register is set.

These two conditions can be accomplished by first writing dma_pending to 0, then waiting until controller_idle is set, then reprogram the timing parameters. If dma_pending was 1 before the

reprogramming started, it should be set again after new timing is in effect to allow the drive to finish the current DMA transfer.

It makes only sense to reprogram the bus timing in the middle of an ongoing DMA transfer when this is necessary because the operating system wants to change the bus clock period. (Dynamic voltage frequency scaling).

It is necessary to wait for controller_idle because a PIO read or write to the ATA bus terminates after the bus cycle with the CPU has been terminated. If the wait for controller_idle does not occur, the new timing values may affect a bus cycle that is still running, and cause error.

The bit iordy_en in register ata_control influences whether the ATA interface will response to the iordy signal coming from the drive. To reprogram it, same rules as for the timing registers apply: Only allowed when dma_pending is cleared, while controller_idle is set.

46.4.3 Access to ATA Bus in PIO Mode

Access to the ATA bus in PIO mode is possible after the following conditions have been met:

- ata_rst_b bit in register ata_control is set.
- Timing parameters have been programmed.

To access the drive in PIO mode, simply read or write to the correct drive register. The bus cycle will be translated to an ATA cycle, and the drive is accessed.

When drive registers are accessed while the ATA bus is in reset, the read or write is discarded, not done.

46.4.4 Using DMA Mode to Receive Data from ATA bus

Apart from PIO mode, the ATA interface supports also MDMA and UDMA mode to transfer data. DMA mode can be used to receive data from the drive (DMA in transfer). In DMA receive mode, the protocol engine will transfer data from the drive to the FIFO using multiword DMA or ultra DMA protocol. The transfer will pause when one of following occurs:

- The FIFO is full.
- The drive deasserts its dma request signal ata_dmarq.
- The bit dma_pending in the ata_contol register is cleared.

When the cause of the transfer pausing is removed, the transfer restarts. The end of the transfer is signalled by the drive to the host by asserting the ata_intrq signal. Alternatively, the host can read the device status register. In this register, the drive will also indicate if the transfer has ended.

The transfer of data from the FIFO into the memory is handled by the host system DMA. Whenever the FIFO filling is above the alarm threshold, the DMA should read one packet of data from the FIFO, and store this in main memory. In doing so, the DMA prevents the FIFO from getting full, and keeps the transfer from drive to FIFO running.

The steps for setting up a DMA data transfer from device to host are:

1. Make sure the ATA bus is not in reset and all timing registers are programmed.
2. Make sure the FIFO is empty by reading it until empty or by resetting it.
3. Initialize the DMA channel connected to fifo_rcv_alarm. Every time fifo_rcv_alarm is high, the DMA should read <packetsize> long ints from the FIFO, and store them to main memory. (typical packetsize is 8 longs)
4. Write 2 × <packetsize> to fifo_alarm register. In this way, FIFO requests attention to DMA when there is at least one packet ready for transfer.
5. To make the ATA ready for a DMA transfer from device to host, take the following steps:
   a) Make sure the FIFO is out of reset by setting bit fifo_rst_b to 1 in the ata control register.
   b) Program fifo_rcv_en=1 in decontrol register. This enables the FIFO to by emptied by the DMA.
   c) Program dma_pending =1, dma_write=0, ultra_mode_selected=0/1 in ata_control register. ultra_mode_selected should be 1 if you want to transfer data using UDMA mode, it should be 0 if you want to transfer data using MDMA mode.
6. Now the host side of the DMA is ready. Send commands to the drive in PIO mode that cause it to request DMA transfer on the ATA bus. The nature of these commands is beyond the scope of this document. Consult the ATA specification to know how to communicate with the drive.
7. When the drive now requests DMA transfer by pulling ata_dmarq high, the ATA interface acknowledges with ata_dmack, and the transfer starts. Data is transferred automatically to the FIFO, and from there on to the host memory.
8. During the transfer, the host can monitor for end of transfer by reading some device ATA registers. These reads will cause the running DMA to pause; after the read is completed, the DMA resumes. The host can also wait unit the drive asserts ata_intrq. This also indicates end of transfer.
9. On end of transfer, the host or host DMA should wait until controller_idle is set, and next read the remaining halfwords from the FIFO, and transfer these to memory.

NOTE

There may be less than <packetsize> remaining bytes, so transfer will not be automatic by the DMA.

46.4.5 Using DMA Mode to Transmit Data to ATA bus

Apart from PIO mode, the ATA interface supports also MDMA and UDMA mode to transfer data. DMA mode can be used to transmit data to the drive (DMA out transfer). In DMA transmit mode, the protocol engine will transfer data from the FIFO to the drive using metalwork DMA or ultra DMA protocol. The transfer will pause when one of following occurs:

- The FIFO is empty.
- The drive deasserts its dma request signal ata_dmarq.
- The bit dma_pending in the ata_contol register is cleared.

When the cause of the transfer pausing is removed, the transfer restarts. The end of the transfer is signalled by the drive to the host by asserting the ata_intrq signal. Alternatively, the host can read the device status register. In this register, the drive will also indicate if the transfer has ended.

The transfer of data from the memory to the FIFO is handled by the host system DMA. Whenever the FIFO filling is below the alarm threshold, the DMA should read one packet of data from the main memory, and store this in the FIFO. In doing so, the DMA prevents the FIFO from getting empty, and keeps the transfer from FIFO to drive running.

The steps for setting up a DMA data transfer from device to host are:

1. Make sure the ATA bus is not in reset and all timing registers are programmed.
2. Make sure the FIFO is empty by reading it until empty, or by resetting it.
3. Initialize the DMA channel connected to fifo_tx_alarm. Every time fifo_tx_alarm is high, the DMA should read <packetsize> long ints from the main memory, and write them to the FIFO. (typical packetsize is 8 longs). Program the DMA such that it will not transfer more than <sectorsize> longwords in total.
4. Write FIFO_SIZE – 2 × <packetsize> to fifo_alarm register. In this way, FIFO will request attention to DMA when there is room for at least one extra packet. FIFO_SIZE should be given in halfwords. (typical 64 halfwords)
5. To make the ATA ready for a DMA transfer from host to device, perform the following steps:
   a) Make sure the FIFO is out of reset by setting bit fifo_rst_b to 1 in the ata control register.
   b) Program fifo_tx_en = 1 in ata_control register. This enables the FIFO to be filled by DMA.
   c) Program dma_pending = 1, dma_write = 1, ultra_mode_selected = 0/1 in ata_control register. ultra_mode_selected should be 1 if you want to transfer data using UDMA mode, it should be 0 if users want to transfer data using MDMA mode.
6. Now, the host side of the DMA is ready. Send commands to the drive in PIO mode that cause it to request DMA transfer on the ATA bus. The nature of these commands is beyond the scope of this document. You should consult the ATA specification to know how to communicate with the drive.
7. When the drive now requests DMA transfer by pulling ata_dmarq high, the ATA interface acknowledges with ata_dmack, and the transfer starts. Data is transferred automatically from the FIFO, and also from host memory to FIFO.
8. During the transfer, the host can monitor for end of transfer by reading some device ATA registers. These reads will cause the running DMA to pause; after the read is completed, the DMA resumes. The host can also wait unit the drive asserts ata_intrq. This also indicates end of transfer.

On end of transfer, no extra FIFO manipulations are needed.

Chapter 47
Pulse-Width Modulator (PWM)

The pulse-width modulator (PWM) has a 16-bit counter and is optimized to generate sound from stored sample audio images. It can also generate tones. It uses 16-bit resolution and a 4 × 16 data FIFO to generate sound.

This section presents an overview of the PWM. Figure 47-1 illustrates the PWM block diagram.

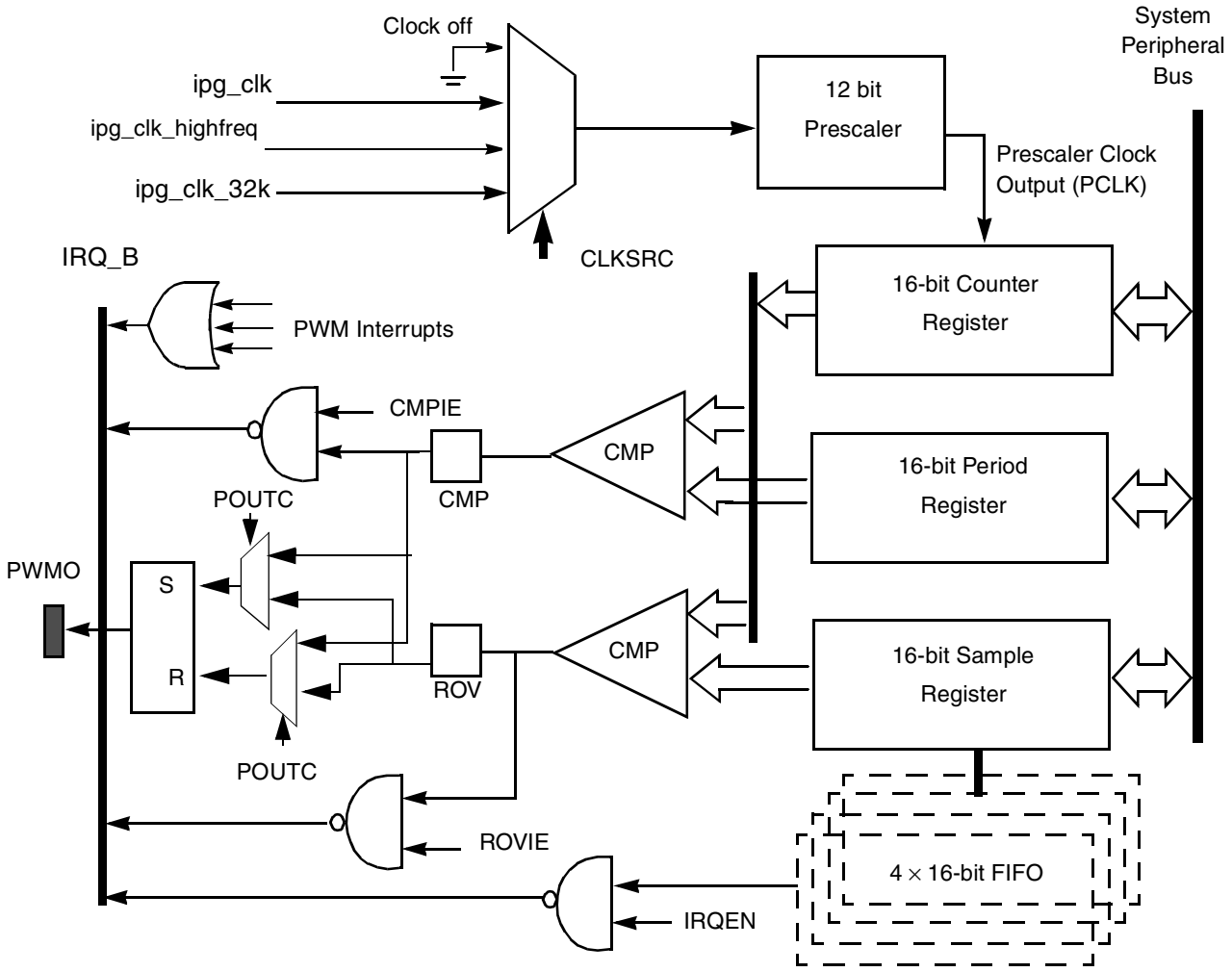


Figure 47-1. Pulse-Width Modulator Block Diagram

The following features characterize the PWM:

- 16-bit up-counter with clock source selection
- 4 × 16 FIFO to minimize interrupt overhead
- 12-bit prescaler for division of clock
- Sound and melody generation
- Active high or active low configured output
- Can be programmed to be active in low power and debug modes
- Interrupts at compare and rollover

47.1 Signal Description

The PWM follows IP Bus protocol for interfacing with the processor core. It does not have any interface signals with any other module inside the chip except for clock and reset inputs from the Clock module and interrupt signals to the processor interrupt handler. There is a single output signal going outside the chip boundary.

47.1.1 External Signals

PWM has a single output signal to the chip boundary named ipp_do_pwmo.

Table 47-1 outlines the external signals.

Table 47-1. External Signals

Name	Direction	Function	Reset State	Pull up
ipp_do_pwmo	Output	This is the functional output of the PWM. The modulated signal of the module is observed at this pin. It can be viewed as a clock signal whose period and duty cycle can be varied with different settings of the PWM. The smallest period can be two ipg_clk clock periods with duty cycle of 50%	0	—

47.2 Memory Map and Register Definition

The PWM module includes six user-accessible 32-bit registers. Section 47.2.2, Register Descriptions," provides the detailed descriptions for all of the PWM registers.

Table 47-2. PWM Memory Map

Offset	Register	Access	Reset Value	Section/Page
0xBASE+0x00 (PWMCR)	PWM Control Register (PWMCR)	R/W	0x0000_0000	47.2.2.1/47-5
0xBASE+0x04 (PWMSR)	PWM Status Register (PWMSR)	R/W	0x0000_0008	47.2.2.2/47-7
0xBASE+0x08 (PWMIR)	PWM Interrupt Register (PWMIR)	R/W	0x0000_0000	47.2.2.3/47-8

Table 47-2. PWM Memory Map (continued)

Offset	Register	Access	Reset Value	Section/Page
0xBASE+0x0C (PWMSAR)	PWM Sample Register (PWMSAR)	R/W	0x0000_0000	47.2.2.4/47-9
0xBASE+0x10 (PWMPR)	PWM Period Register (PWMPR)	R/W	0x0000_FFFE	47.2.2.5/47-10
0xBASE+0x14 (PWMCNR)	PWM Counter Register (PWMCNR)	R	0x0000_0000	47.2.2.6/47-11

47.2.1 Register Summary

Figure 47-2 shows the key to the register fields and Table 47-3 shows the register figure conventions.

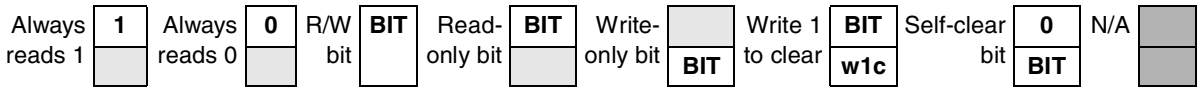


Figure 47-2. Key to Register Fields

Table 47-3. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 47-4 shows the PWM register summary.

Table 47-4. PWM Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x00 (PWMCR)	R	0	0	0	0	FWM		STOPEN	DOZEN	WAITEN	DBGEN	BCTR	HCTR	POUTC		CLKSRC	
	W																
	R	PRESCALER												SWR	REPEAT		EN
	W																
0xBASE+0x04 (PWMSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	FWE	CMP	ROV	FE	FIFOAV		
	W										w1c	w1c	w1c	w1c			
0xBASE+0x08 (PWMIR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	CIE	RIE	FIE
	W																
0xBASE+0x0C (PWMSAR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	SAMPLE[15:0]															
	W																
0xBASE+0x10 (PWMPR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	PERIOD[15:0]															
	W																
0xBASE+0x14 (PWMCNR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	COUNT[15:0]															
	W																

47.2.2 Register Descriptions

This section contains the detailed register descriptions for the PWM registers.

47.2.2.1 PWM Control Register (PWMCR)

The PWM control register (PWMCR), shown in Figure 47-3, is used to configure the operating settings of the PWM. It contains the prescaler for the clock division.

0xBASE+0x00 (PWMCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	FWM		STOP EN	DOZ EN	WAIT EN	DBG EN	BCT R	HCT R	POUTC		CLKSRC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALER												SWR	REPEAT		EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 47-3. PWM Control Register (PWMCR)

Table 47-5 shows the PWMCR field descriptions.

Table 47-5. PWMCR Field Descriptions

Field	Description
31–28 Reserved	Reserved. These reserved bits are always read as zero.
27–26	FIFO Water Mark These bits are used to set the data level at which the FIFO empty flag will be set and the corresponding interrupt generated. 00 FIFO empty flag is set when there are more than or equal to 1 empty slots in FIFO 01 FIFO empty flag is set when there are more than or equal to 2 empty slots in FIFO 10 FIFO empty flag is set when there are more than or equal to 3 empty slots in FIFO 11 FIFO empty flag is set when there are more than or equal to 4 empty slots in FIFO
25 STOPEN	Stop Mode Enable This bit keeps the PWM functional while in stop mode. When this bit is cleared, the input clock is gated off in stop mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in stop mode 1 Active in stop mode
24 DOZEN	Doze Mode Enable This bit keeps the PWM functional in doze mode. When this bit is cleared, the input clock is gated off in doze mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in doze mode 1 Active in doze mode

Table 47-5. PWMCR Field Descriptions (continued)

Field	Description
23 WAITEN	Wait Mode Enable This bit keeps the PWM functional in wait mode. When this bit is cleared, the input clock is gated off in wait mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in wait mode 1 Active in wait mode
22 DBGEN	Debug Mode Enable This bit keeps the PWM functional in debug mode. When this bit is cleared, the input clock is gated off in debug mode. This bit is not affected by software reset. It is cleared by hardware reset. 0 Inactive in debug mode 1 Active in debug mode
21 BCTR	Byte Data Swap Control This bit determines the byte ordering of the 16-bit data when it goes into the FIFO from the sample register. 0 byte ordering remains the same 1 byte ordering is reversed
20 HCTR	Half-word Data Swap Control This bit determines which half word data from the 32 bit IP-Bus interface is written into the lower 16 bits of the sample register. 0 Half word swapping does not take place 1 Half words from write data bus are swapped
19–18 POUTC	PWM Output Configuration This bit field determines the mode of PWM output on the output pin. 00 Output pin is set at rollover and cleared at comparison 01 Output pin is cleared at rollover and set at comparison 10 PWM output is disconnected 11 PWM output is disconnected
17–16 CLKSRC	Select Clock Source These bits determine which clock input will be selected for running the counter. After reset the system functional clock is selected. The input clock can also be turned off if these bits are set to 00. This field value should only be changed when the PWM is disabled 00 Clock is off 01 ipg_clk 10 ipg_clk_highfreq 11 ipg_clk_32k
15 –4 PRESCALER	Counter Clock Prescaler Value This bit field determines the value by which the clock will be divided before it goes to the counter. 0x000 Divide by 1 0x001 Divide by 2 ... 0xfff Divide by 4096
3 SWR	Software Reset PWM is reset when this bit is set to 1. It is a self clearing bit. A write 1 to this bit is a single wait state write cycle. When the module is in reset state this bit is set and is cleared when the reset procedure is over. Setting this bit resets all the registers to their reset values except for the STOPEN, DOZEN, WAITEN, and DBGEN bits in this control register. 0 PWM is out of reset 1 PWM is undergoing reset

Table 47-5. PWMCR Field Descriptions (continued)

Field	Description
2–1 REPEAT	Sample Repeat This bit field determines the number of times each sample from the FIFO is to be used. 00 Use each sample once 01 Use each sample twice 10 Use each sample four times 11 Use each sample eight times
0 EN	PWM Enable This bit enables the PWM. If this bit is not enabled, the clock prescaler and the counter is reset. When the PWM is enabled, it begins a new period, the output pin is set to start a new period while the prescaler and counter are released and counting begins. 0 PWM disabled 1 PWM enabled

47.2.2.2 PWM Status Register (PWMSR)

The PWM status register (PWMSR), shown in Figure 47-4, contains seven bits which display the state of the FIFO and the occurrence of rollover and compare events. The FIFOAV bit is read-only but the other four bits can be cleared by writing 1 to them. FE, ROV, and CMP bits are associated to Fifo-Empty, Roll-over, and Compare interrupts, respectively.

0xBASE+0x04 (PWMSR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	FWE	CMP	ROV	FE	FIFOAV		
W										w1c	w1c	w1c	w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Figure 47-4. PWM Status Register (PWMSR)

Table 47-6 shows the PWMSR field descriptions.

Table 47-6. PWMSR Field Descriptions

Field	Description
31–7 Reserved	Reserved. These reserved bits are always read as zero.
6 FWE	FIFO Write Error Status. This bit shows that an attempt has been made to write FIFO when it is full. 0 FIFO write error not occurred 1 FIFO write error occurred

Table 47-6. PWMSR Field Descriptions (continued)

Field	Description
5 CMP	Compare Status. This bit shows that a compare event has occurred. 0 Compare event not occurred 1 Compare event occurred
4 ROV	Roll-over Status. This bit shows that a roll-over event has occurred. 0 Roll-over event not occurred 1 Roll-over event occurred
3 FE	FIFO Empty Status Bit. This bit indicates the FIFO data level in comparison to the water level set by FWM field in the control register. 0 Data level is above water mark 1 When the data level falls below the mark set by FWM field
2–0 FIFOAV	FIFO Available. These read-only bits indicate the data level remaining in the FIFO. An attempted write to these bits will not affect their value and no transfer error is generated. 000 No data available 001 1 word of data in FIFO 010 2 words of data in FIFO 011 3 words of data in FIFO 100 4 words of data in FIFO 101 Unused 110 Unused 111 Unused

47.2.2.3 PWM Interrupt Register (PWMIR)

The PWM Interrupt register (PWMIR), shown in Figure 47-5, contains three bits which control the generation of the compare, rollover and FIFO empty interrupts.

0xBASE+0x08 (PWMIR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	CIE	RIE	FIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 47-5. PWM Interrupt Register (PWMIR)

Table 47-7 shows the PWMIR field descriptions.

Table 47-7. PWMIR Field Descriptions

Field	Description
31–3 Reserved	Reserved. These reserved bits are always read as zero.
2 CIE	Compare Interrupt Enable. This bit controls the generation of the Compare interrupt. 0 Compare Interrupt not enabled 1 Compare Interrupt enabled
1 RIE	Roll-over Interrupt Enable. This bit controls the generation of the Rollover interrupt. 0 Roll-over interrupt not enabled 1 Roll-over Interrupt enabled
0 FIE	FIFO Empty Interrupt Enable. This bit controls the generation of the FIFO Empty interrupt. 0 FIFO Empty interrupt disabled 1 FIFO Empty interrupt enabled

47.2.2.4 PWM Sample Register (PWMSAR)

The PWM sample register (PWMSAR), shown in Figure 47-6, is the input to the FIFO. 16-bit words are loaded into the FIFO. The FIFO can be written and read when the PWM is disabled. The PWM will run at the last set duty-cycle setting if all the values of the FIFO has been utilized, until the FIFO is reloaded or the PWM is disabled. When a new value is written, the duty cycle changes after the current period is over.

A value of zero in the sample register results in the ipp_pwm_pwmo output signal being always low/high (POUTC = 00 is low and POUTC = 01 is high), and hence no output waveform will be produced. If the value in this register is higher than the PERIOD + 1, the output will never be reset/set depending on POUTC value.

0xBASE+0x0C (PWMSAR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SAMPLE															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 47-6. PWM Sample Register (PWMSAR)

Table 47-8 shows the PWMSAR field descriptions.

Table 47-8. PWMSAR Field Descriptions

Field	Description
31–16 Reserved	These are reserved bits and writing a value will not affect the functionality of PWM and are always read as zero.
15–0 SAMPLE	Sample Value. This is the input to the 4x16 FIFO. The value in this register denotes the value of the sample being currently used.

47.2.2.5 PWM Period Register (PWMPR)

The PWM period register (PWMPR), shown in Figure 47-7, determines the period of the PWM output signal. After the counter value matches PERIOD + 1, the counter is reset to start another period.

PWMO (Hz) = PCLK(Hz) ÷ (period + 2)

A value of zero in the PWMPR will result in a period of two clock cycles for the output signal. Writing 0xFFFF to this register will achieve the same result as writing 0xFFFE.

A change in the period value due to a write in PWMPR results in the counter being reset to zero and the start of a new count period.

0xBASE+0x10 (PWMPR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIOD															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Figure 47-7. PWM Period Register (PWMPR)

Table 47-9 shows the PWMPR field descriptions.

Table 47-9. PWMPR Field Descriptions

Field	Description
31–16 Reserved	These are reserved bits and writing a value will not affect the functionality of PWM and are always read as zero.
15–0 PERIOD	Period Value. These bits determine the Period of the count cycle. The counter counts up to [Period Value] +1 and is then reset to 0x0000.

47.2.2.6 PWM Counter Register (PWMCNR)

The read-only pulse-width modulator counter register (PWMCNR), shown in Figure 47-8, contains the current count value and can be read at any time without disturbing the counter.

0xBASE+0x14 (PWMCNR) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 47-8. PWM Counter Register (PWMCNR)

Table 47-10 shows the PWMCNR field descriptions.

Table 47-10. PWMCNR Field Descriptions

Field	Description
31–16 Reserved	These are reserved bits and writing a value will not affect the functionality of PWM and are always read as zero.
15–0 COUNT	Counter Value. These bits are the counter register value and denotes the current count state the counter register is in.

47.3 Functional Description

The following sections detail the PWM operation and function.

47.3.1 Operation

The output of the PWM is a toggling signal whose frequency and duty cycle can be modulated by programming the appropriate registers. It has a 16-bit up counter which counts from 0x0000 until the counter value equals the [Value in Period register] + 1. After this match occurs the counter is reset to 0x0000.

At the beginning of a count period cycle, the PWMO pin is set to one (default) and the counter begins counting up from 0x0000. The sample value in the sample FIFO is compared on each count of prescaler clock. When the sample and count values match, the PWMO signal is cleared to zero (default). The counter continues counting until the period match occurs and subsequently another period cycle begins.

When the PWM is enabled the counter starts running and generates an output with the reset values in the period and sample registers. It is recommended that the programming of these registers be done before PWM is enabled.

A hardware reset results in all the PWM count and sample registers begin cleared and the FIFO being flushed. The control register shows that FIFO is empty and it can be written into, and the PWM is disabled. A software reset has the same results, however the state of the STOPEN, DOZEN, WAITEN, and DBGEN bits in the control register are not affected. Software reset can be asserted even when the PWM is in disabled state.

47.3.1.1 Clocks

The clock that feeds the prescaler can be selected from:

- **High frequency Clock** (ipg_clk_highfreq) pat_ref or ckih

  This is a high frequency clock provided by the Clock Module (CM). This clock is supposed to be on in the low power mode when the ipg_clk is turned off. Thus the PWM can be run on this clock in the low power mode. The CRM is expected to provide this clock after synchronizing it to ahb_clk in the normal functional mode and switch to the unsynchronized version in the low power mode.
- **Low Reference Clock** (ipg_clk_32k) ckil

  This is the 32 KHz low reference clock which is provided by the CM. This clock is supposed to be on in the low power mode when ipg_clk is turned off. Thus PWM can be run on this clock in the low power mode. The CRM is expected to provide this clock after synchronizing it to ahb_clk in the normal functional mode and switch to the unsynchronized version in the low power mode.
- **Global Functional Clock** (ipg_clk)

  This clock is supposed to be on in normal operations. In low power modes it can be switched off.

The clock input source is determined by the CLKSRC field of the PWM control register. **The CLKSRC value should only be changed when the PWM is disabled.**

A change in the value of the PRESCALER field of the control register is immediately reflected on its output clock frequency.

47.3.1.2 FIFO

Digital sample values can be loaded into the pulse-width modulator as 16-bit words. The endianess can be changed using the BCTR and HCTR bits of the control register. A 4-word (16-bit) FIFO minimizes interrupt overhead. A maskable interrupt is generated when the number of data words fall below the water level set by the FWM field in the control register.

A write in the sample register results in the value being stored into the FIFO if it is not full. A write when the FIFO is full sets FWE (FIFO write error) bit in the status register and the FIFO contents remain unchanged. The FIFO can be written into when the PWM is disabled. The FIFOAV field shows how many data words are currently contained in the FIFO and if it can be written into.

A read on the sample register yields the current FIFO value being used or will be used by the PWM for generation on the output signal. Therefore a write and a subsequent read on the sample register may result in different values being obtained.

47.3.1.3 Rollover and Compare Event

The counter is reset to 0x0000 after its value equals the PERIOD + 1 and resumes counting thereafter. This event is referred to as a rollover. When PERIOD = 0x0000, the counter is reset after count reaches 0x0001. Therefore PERIOD = 0xFFFF or 0xFFFE results in the counter value being reset after count until 0xFFFF. During a rollover event the output is either set (default), reset or has no effect according to the programming of the POUTC field in the control register. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

When the counter value reaches the sample value, the output of the PWM is reset (default), set, or has no effect according to the programming of the POUTC field of control register. This event is referred to as a compare event. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

If the rollover event sets the PWM output signal the compare event will reset it and vice versa for a particular programming configuration of POUTC field.

47.3.1.4 Low Power Mode Behavior

In low power modes if the clock from the selected clock source is available, the PWM counter continues to run and an output is produced depending on whether the control bit for that mode is set. In the absence of the clock itself or if the corresponding low power bit in the control register is 0, the counter is reset and resumes counting when it exits the low power mode.

47.3.1.5 Debug Mode Behavior

In debug mode, PWM has the option of continuing to run or be halted. If the DBGEN bit is not set in the PWMCR, the PWM is halted. If the DBGEN bit is set, then the PWM will continue to run in the debug mode.

Chapter 48
Run-Time Integrity Checker (RTIC)

The Run-Time Integrity Checker (RTIC) function is to ensure the integrity of the peripheral memory contents and assist with boot authentication. The RTIC can verify memory contents during system boot and during runtime execution. If the memory contents at runtime fail to match the hash signature, an error in the security monitor is triggered.

Figure 48-1 is a block diagram of the RTIC.

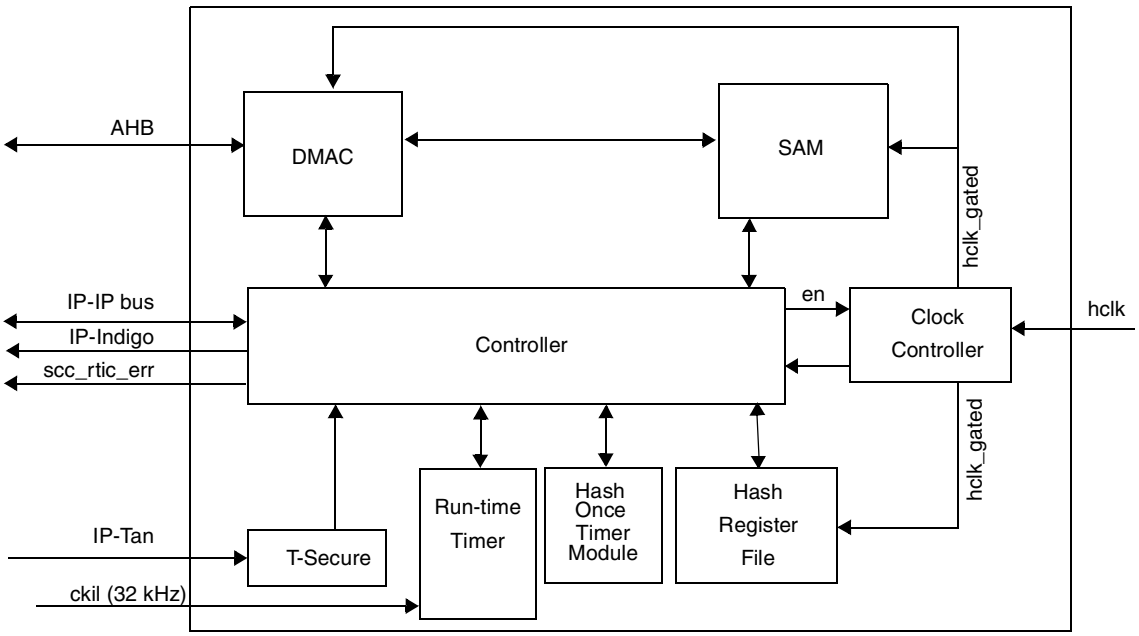


Figure 48-1. RTIC Block Diagram

48.1 Features

The RTIC offers the following features:

- SHA-1 message authentication
- Input DMA interface
- Segmented data gathering to support non-contiguous data blocks in memory (up to two segments per block)
- Works with high assurance boot process
- Support for up to four independent memory blocks
- Programmable DMA bus duty cycle timer and watchdog timer

- Power-saving clock gating logic
- Hardware configurable big/little-endian data format
- Full word memory reads (word-aligned addresses, multiple of 32-bit lengths)

48.1.1 Modes of Operation

The RTIC operates in two primary modes:

- One-time hash mode
  - Is used during high assurance boot for code authentication or one time integrity checking
  - Stores hash result internally and signals interrupt to host
- Continuous hash mode
  - Is used at run-time to continuously to verify integrity of memory contents
  - Checks re-generated hash against internally stored values and interrupts host only if error occurs

48.2 Initialization/Application Information

The RTIC serves as a single-use hash accelerator to assist with code authentication and other services at boot time as well as an autonomous/passive memory integrity checker during runtime. It is programmed through the IP-slave interface and scans the peripheral memory contents over the AHB interface using direct memory access.

Figure 48-2 shows a typical system configuration using the RTIC.

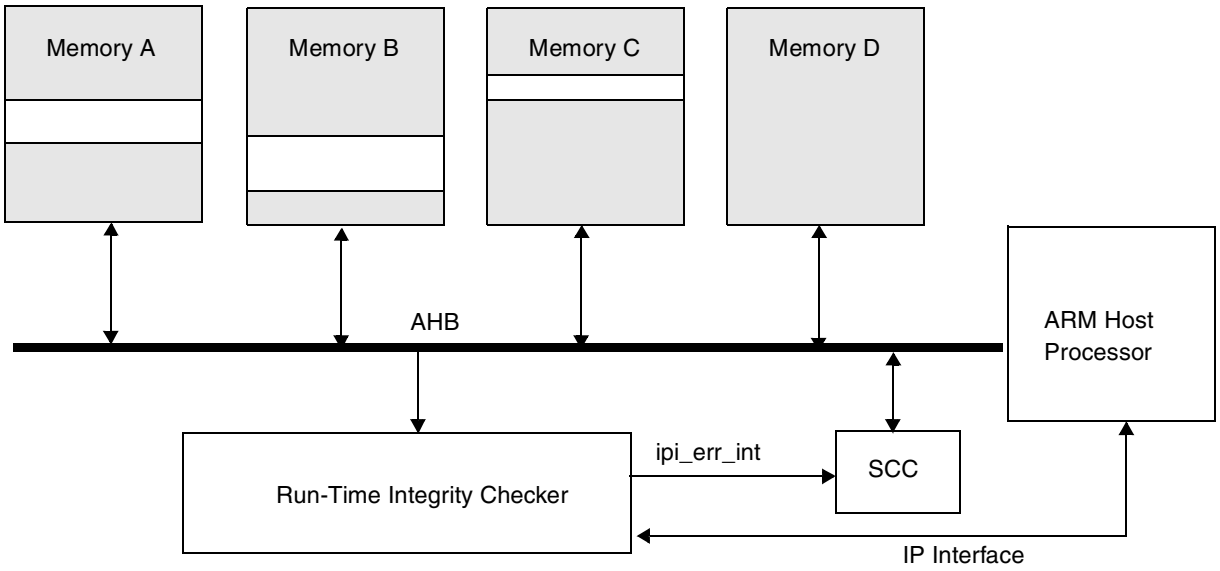


Figure 48-2. System Diagram

In this example, there are four independent memory blocks that can be checked by the RTIC. Memories A,B, and C have their contents partitioned over non-contiguous spaces. Memory D does not contain any physical partitioning. The host program sthe RTIC with the starting address and length of each partition

inside memory A, B, and C. For memory D, only one starting address and length is specified, with the second start address/length fields for memory D being set to 0.

After setting the A/B/C/D hash once memory enable bits in the RTIC control register and hash once bit in the RTIC command register, the RTIC hashes each memory and stores the result in its hash register file to be read by the host. If the RTIC is used to verify that the memories are not corrupted during runtime, the A/B/C/D run-time memory enable bits in the control register must be set, followed by the runtime check bit in the RTIC command register. The RTIC rehashes each enabled memory in a continuous loop until either an error occurs or the RTIC is reset.

NOTE

Detailed information about the RTIC is provided in the i.MX51 Security Manual. Contact your Freescale representative for information about obtaining this document.

Chapter 49
Symmetric/Asymmetric Hashing and Random Accelerator (SAHARA)

The symmetric/asymmetric hashing and random accelerator (SAHARA) is a security coprocessor that can be used on cell phone baseband processors or wireless PDAs. It implements block encryption algorithms, (AES, DES, and 3DES), hashing algorithms (MD5, SHA-1, SHA-224, and SHA-256), a stream cipher algorithm (ARC4), and a hardware random number generator. It has a slave IP bus interface that the host can use to write configuration and command information and to read status information. It also has a DMA controller with an AHB bus interface that reduces the burden on the host to move the required data to and from memory.

See Figure 49-1 for the block diagram of SAHARA.

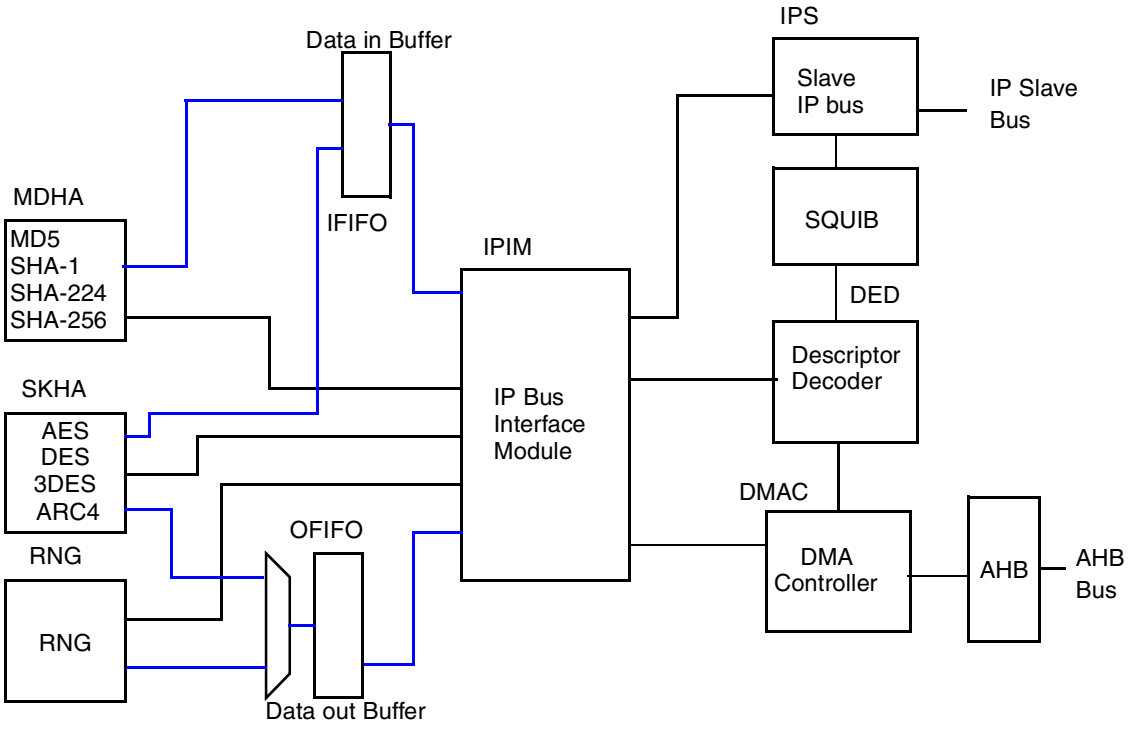


Figure 49-1. Block Diagram of SAHARA

49.1 Features

SAHARA accelerates the following security functions:

- AES encryption/decryption
  - — ECB, CBC, CTR, and CCM modes
  - — 128-bit key
- DES/3DES
  - — ECB, CBC, and CTR modes
  - — 56-bit key with or without parity (DES)
  - — 112-bit or 168-bit key with parity (3DES)
- ARC4 (RC4-compatible cipher)
  - – 5-16 byte key
  - – Host accessible S-box
- MD5, SHA-1, SHA-224 and SHA-256 hashing algorithms.
  - — Messages lengths that are multiples of bytes (with auto padding).
  - — Auto padding supported.
  - — HMAC (support for IPAD and OPAD via Descriptors).
  - — Up to 2^{32} byte message length (with auto padding).
- Pseudorandom number generator
  - — Entropy is generated via independent free running ring oscillators

SAHARA also provides the following features:

- Descriptor based processing to reduce communication between host processor and SAHARA
- Multiple Master Interface (SQUIB), allows two masters to share SAHARA transparently.
- ARM TrustZone support via additional AHB and IPBus signalling
- Low power design
  - — Automatic power down of individual blocks when not in use
  - — Clock gating on registers
  - — RNG sleep mode
- Restricted access to potentially sensitive information
  - — Internal registers are cleared after a Descriptor chain has completed processing in BATCH mode.
  - — Security Monitor can cause data to be cleared.
  - — Scan reset and scan exit signals prevent data being scanned out.
- Mixed endianness support.
- Optimization for maximum of 8-word AHB burst

49.1.1 Modes of Operation

SAHARA can be used in three modes: batch, dedicated, and debug. The mode bits can be changed only when SAHARA is idle (that is, when the status register busy bit is 0). Otherwise, the mode bits are not written and no other indication is given. Debug mode can be entered only when the secure state bit in the status register is 0, which reflects the SCC secure state input (typically from the SCC module).

49.1.1.1 Batch Mode

Batch mode is the default operating mode. After a chain of descriptors has been processed, the internal functional blocks are initialized and an interrupt is generated. Initializing SAHARA prevents a task from being able to access the left over data from a descriptor chain of a different task.

49.1.1.2 DedicateD Mode

Dedicated mode is used when SAHARA is dedicated to a single task. In dedicated mode, SAHARA does not reset internal state information at the end of a descriptor chain. This allows a task to improve performance by not having to save and restore its context information between each set of descriptor chains. This is useful when a task does not have access to all data at one time, and must therefore present its descriptors at different times.

49.1.1.3 Debug Mode

In direct access debug mode, the host writes directly to the address space of the internal hardware blocks. SAHARA single-steps descriptors only when in debug mode. This is intended only for interactive debugging and is disabled when the Security Monitor is in the secure state (that is, the scc_secure_state input is asserted).

In debug mode, the host processor can read and write the internal encryption engine's registers. It is up to the host processor to set up the internal flow control by writing the appropriate value in the internal flow control register. If this is not set up properly, then when attempting to get a module to read data from the input buffer, the module will probably generate an error and require resetting. No interrupts are generated while SAHARA is in debug mode.

The processing of a descriptor can be traced in debug mode by writing to the Single_Step bit in the command register. This causes the descriptor decoder to perform a single operation of descriptor processing, and then allows the host to monitor the internal registers to see the result of that operation. Writing to the Single_Step bit in the command register multiple times causes the descriptor processing to proceed until it completes. If two single-step commands arrive too quickly, the second is simply ignored. During the single step processing, the status register state bits indicate busy. When the processing completes, the state reverts back to idle.

When single stepping, the internal flow control register is modified by the descriptor decoder. The host should not change it or an error may result.

NOTE

Detailed information about the SAHARA is provided in the i.MX51 Security Manual. Contact your Freescale representitive for information about obtaining this document.

Chapter 50
Security Controller (SCC)

The Security Controller (SCC) is composed of two sub-blocks, secure RAM, and a security monitor (see Figure 50-1).

The primary functionality of the SCC is to establish the following:

- A centralized security state controller and hardware security state with a hardware-configured, unalterable security policy
- An uninterruptible hardware mechanism that detects and responds to threat detection signals (specifically, platform test access signals)
- A device-unique data protection/encryption resource that enables off-chip storage of security-sensitive data
- An internal storage resource that automatically and irrevocably destroys plain text security sensitive data upon threat detection

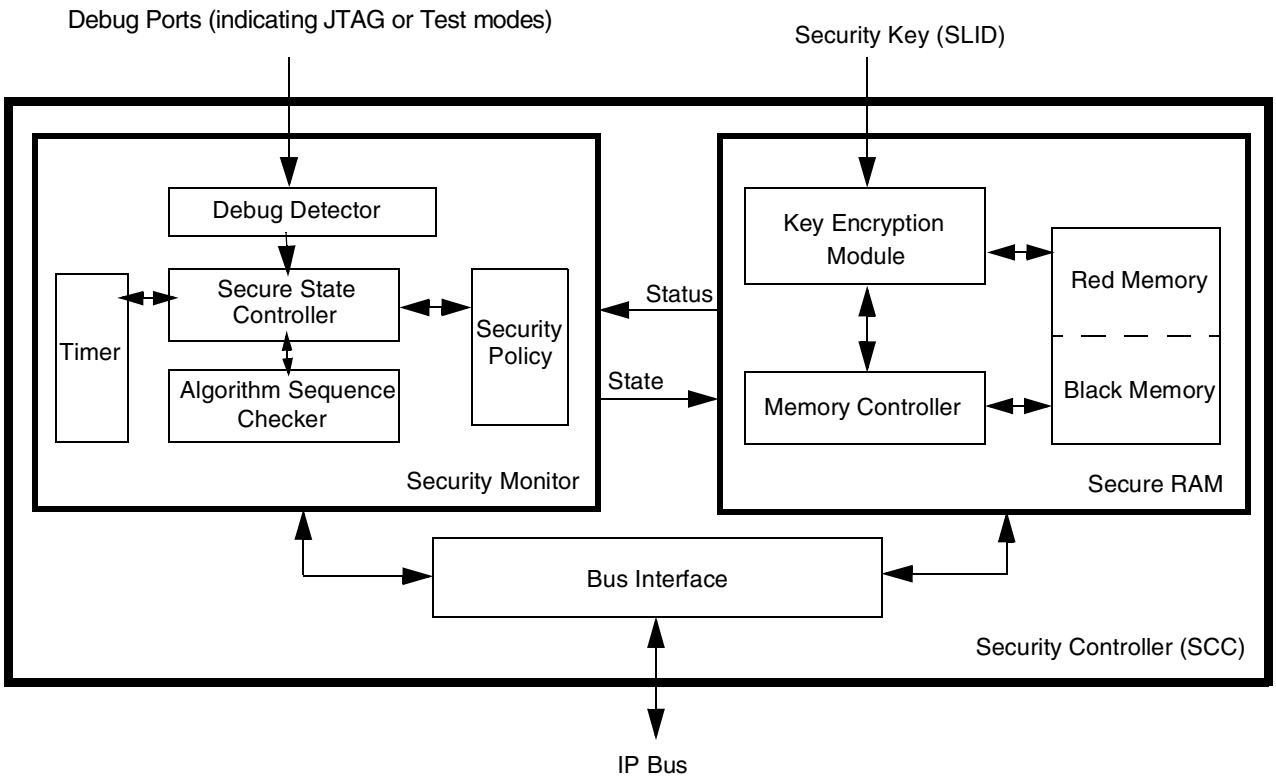


Figure 50-1. Security Controller Block Diagram

50.1 Overview

Security and security services, in an embedded or data processing platform, refer to the platform's ability to provide mandatory and optional information protection services. Information in this context refers to all embedded data, both program store and data load. Therefore, a secure platform is intended to protect information/data from unauthorized access in the form of inspection (read), modification (write), or execution (use).

NOTE

Detailed information about the SCC is provided in the i.MX51 Security Manual. Contact your Freescale representitive for information about obtaining this document.

Chapter 51
Subscriber Identification Module (SIM)

The subscriber identification module (SIM) is designed to facilitate communication to SIM cards or Eurochip pre-paid phone cards. The SIM module has two ports that can be used to interface with the various cards. See the SIM block diagram in Figure 51-1.

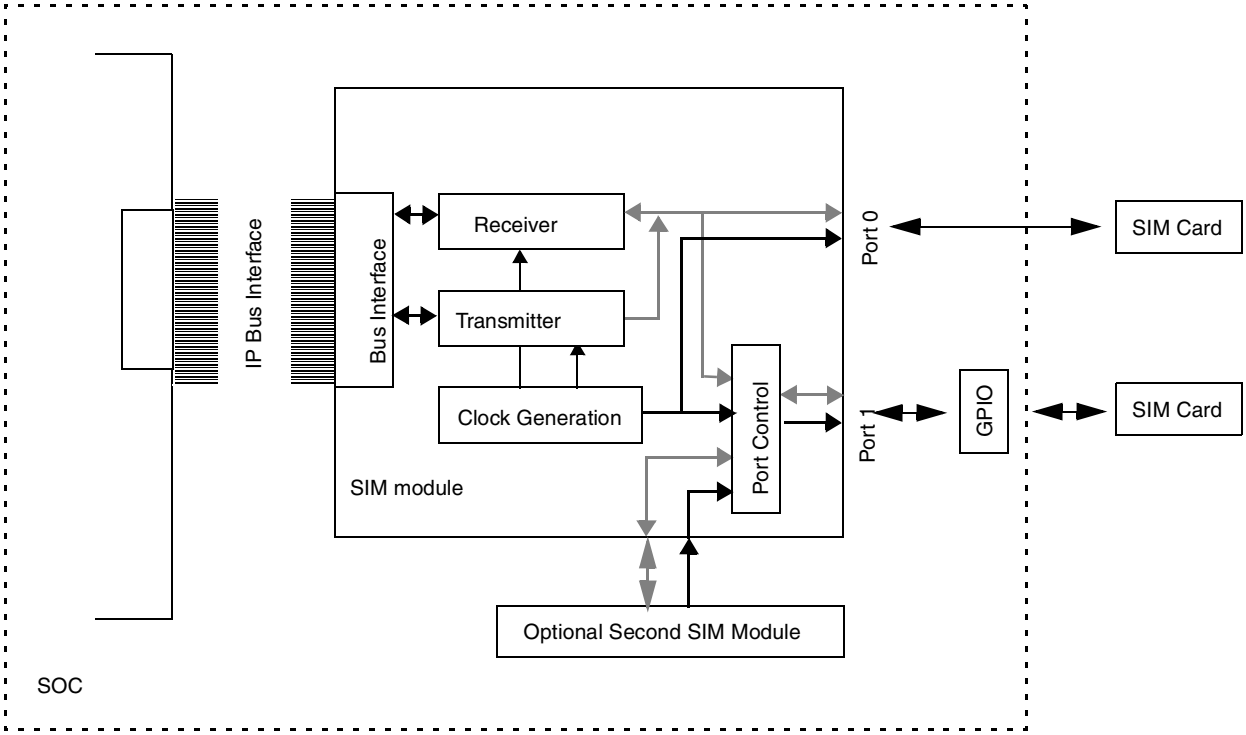


Figure 51-1. SIM Block Diagram

51.1 Overview

The SIM design is summarized in the following sections.

- Section 51.1.1, “Modes of Operation”
- Section 51.1.2, “SIM Bus Interface Overview”
- Section 51.1.3, “SIM Clock Generator Overview”
- Section 51.1.4, “SIM Transmitter Overview”
- Section 51.1.5, “SIM Receiver Overview”
- Section 51.1.6, “SIM Port Control Overview”
- Section 51.1.7, “SIM General Purpose Counter Overview”
- Section 51.1.8, “SIM LRC Block Overview”
- Section 51.1.9, “SIM CRC Block Overview”

51.1.1 Modes of Operation

The SIM module I/O interface can be operated in one of three modes of operation, as summarized below.

- Two wire interface

  In this mode both the IC pin RX and TX are used to interface to the SmartCard. This is activated by resetting the 3VOLTx bit in the port control register to a “0”.

- External one wire interface

  In this mode the IC pins RX and TX are tied together external to the IC and routed to the SmartCard. The 3VOLT bit in the port control register is reset to a “0” and the OD_Px bit in the OD_CONFIG register is set to a “1”. For this interface to work properly the IC pin (RX-TX) must be pulled high by a resistor. The value should be selected small enough to give a fast enough rise time.

- Internal one wire interface

  In this mode the IC pin TX is routed to the SmartCard. The receive pin RX is connected to the TX pin internal to the IC. The 3VOLTx bit in the port control register is reset to a “1” and the OD_Px bit in the OD_CONFIG register is set to a “1”. For this interface to work properly the IC pin TX must be pulled high by a resistor. The value should be selected small enough to give a fast enough rise time.

CAUTION

For the internal one wire interface to work, the bond pad must be capable of simultaneous input and output. The input path should not be disabled when the output is enabled.

51.1.2 SIM Bus Interface Overview

The SIM module is designed with a separate block that encompasses the interface to the IP Bus. The bus interface is built in such a way as to be easily modified to other bus definitions. The SIM module is to be considered a 32-bit peripheral. To avoid endian issues with register access, all read/writes should be done as word access(32 bit). The bus interface has the following features:

- Peripheral Address/chip select decoding
- Illegal Address generation
- Dynamic wait state generation (tied inactive)
- Register organization
- Register bit implementations

See Table 51-1 for summary of SIM top level interrupts.

Table 51-1. SIM Top Level Interrupt Summary

Flag	Interrupt Sources	Interrupt Masks	Description
SIMIRQ_N	XTE, OEF, SDI0, SDI1, TFO, GPCNT, BWT, BGT, CWT, RTE	XTM, OIM, SDIM0, SDIM1, TFOM, GPCNTM, BWTM, BGTM, CWTM, RTM	SIM General Interrupt
SIMDAT_N	TC, ETC, TFE, RDRF, TDTF, RFE	TCIM, ETCIM, TFEIM, RIM, TDTFM,RFEM	SIM Data Interrupt

51.1.3 SIM Clock Generator Overview

The SIM module contains a block designed specifically for generating the clocks used internal to the SIM module, and the clocks provided to the SIM cards. The clock generator has the following features:

- Programmable Divisor (CLK_PRESCALER[7:0]) for SIM card clock generation (***45%~55% duty cycle except for divider by 3, 5, 7, 9***)
- Receive clock generation (/1, /2, /4, /8, /16, /32, /31) from first stage
- Transmit clock generation (/12, /8) from second stage
- Programmable Divisor for Receive clock generation
- There are no interrupt sources generated by the SIM clock generator block

51.1.4 SIM Transmitter Overview

The SIM transmitter block contains the transmit state machine, transmit shift register, and transmit FIFO. The transmitter has the following features.

- 16 bytes deep transmit FIFO
- Automatic NACK generation on parity and overrun errors
- Hardware data format support (inverse convention or direct convention)
- Retransmission of data upon SIM card NACK request with programmable maximum threshold of retransmissions
- Programmable guard time between transmitted bytes
- Six interrupt sources (see Table 51-2)

Table 51-2. SIM Transmitter Interrupt Summary

Flag	Flag Register	Mask	Mask Register	Description
TC	XMT_STATUS	TCIM	INT_MASK	Transmit Complete
ETC	XMT_STATUS	ETCIM	INT_MASK	Early Transmit Complete
TFE	XMT_STATUS	TFEIM	INT_MASK	Transmit FIFO Empty
XTE	XMT_STATUS	XTM	INT_MASK	Transmit Threshold Error
TFO	XMT_STATUS	TFOM	INT_MASK	Transmit FIFO Overfill Error
TDTF	XMT_STATUS	TDTFM	INT_MASK	Transmit Data Threshold Flag

51.1.5 SIM Receiver Overview

The SIM Receiver block contains the receive state machine, receive FIFO, and control logic. The receiver has the following features.

- 288 bytes deep receive FIFO
- Decoding of initial character mode for setting data format
- Hardware data format support (inverse convention or direct convention)
- NACK detection

- 11 ETU character support
- Character Wait Time Counter
- Six interrupt sources (see Table 51-3)

Table 51-3. SIM Receiver Interrupt Summary

Flag	Flag register	Mask	Mask register	Description
BGT	RCV_STATUS	BGTM	INT_MASK	Block Guard Time flag
BWT	RCV_STATUS	BWTM	INT_MASK	Block Wait Time flag
RTE	RCV_STATUS	RTM	INT_MASK	Receive Nack threshold
CWT	RCV_STATUS	CWTM	INT_MASK	Character Wait Time flag
OEF	RCV_STATUS	OIM	INT_MASK	Overrun Error Flag
RDRF	RCV_STATUS	RIM	INT_MASK	Receive Data Register Full
RFE	RCV_STATUS	RFEM	INT_MASK	Receive Fifo Empty

51.1.6 SIM Port Control Overview

The SIM module supports numerous port control functions necessary for SIM card interaction. Each of the two SIM card ports has the following I/O: CLK, $\overline{\text{RST}}$, DATA_RCV, DATA_XMT, SIMPD (SIM Presence Detect), and SVEN (SIM Vcc enable). The following list gives a number of the important features of the port logic.

- Open-drain or push pull DATA_XMT pin
- Port 1, port 0, or alternate port selection
- SIM card presence detect with interrupt capability
- Deep sleep wake-up via SIM card presence detect interrupt
- Manual control of all SIM card interface signals
- Automatic power down of port logic on SIM card presence detect
- Two interrupt sources (see Table 51-4)

Table 51-4. SIM Card Detect Interrupts

Flag	Flag register	Mask	Mask Register	Description
SDI1	PORT1_DETECT	SDIM1	PORT1_DETECT	SIM Detect Interrupt for port 1
SDI0	PORT0_DETECT	SDIM0	PORT0_DETECT	SIM Detect Interrupt for port 0

51.1.7 SIM General Purpose Counter Overview

The SIM module provides a 16-bit counter that can be configured to count using clock sources from the card clock, receive clock, or transmitter clock (ETU Clock). A programmable 16-bit register is provided to compare with the counter for interrupt generation:

- Selectable clock source
- 16-bit counter and comparator

- One Interrupt source (see Table 51-5)

Table 51-5. SIM General Purpose Counter Interrupts

Flag	Flag Register	Mask	Mask Register	Description
GPCNT	XMT_STATUS	GPCNTM	INT_MASK	GPCNT Comparator Interrupt

51.1.8 SIM LRC Block Overview

The SIM module contains a block designed to generate linear redundancy checking (LRC) information for both received and transmitted characters. The LRC block has the following features:

- 8-bit LRC value
- Valid LRC Detector
- There are no interrupt sources generated by the SIM LRC block

51.1.9 SIM CRC Block Overview

The SIM module contains a block designed to generate cyclic redundancy checking (CRC) information for both received and transmitted characters. The CRC block has the following features:

- 16-bit CRC value
- 16-bit CRC Polynomial Generator
- Valid CRC Detector
- There are no interrupt sources generated by the SIM CRC block

51.2 External Signal Description

51.2.1 Overview

See Table 51-6 for summary of the SIM module signals.

Table 51-6. Signal Properties

Name	Port	Function	Reset State	Pull-up
External Signals (Pads)				
sim_pin_sclk0	out	Clock for the smartcard on port0	0	Active
sim_pin_srst0	out	Reset signal for port0	0	Active
sim_pin_sven0	out	Vcc enable signal for port0	0	Active
sim_pin_data0_tx_out	out	Transmit data signal for port0	0	Active/Passive
pin_sim_rcvd0_in	in	Receive data signal for port0	—	—
pin_sim_simpd0	in	Card insertion detect signal for port0	—	—
sim_pin_sclk1	out	Clock for the smartcard on port1	0	Active

Table 51-6. Signal Properties (continued)

Name	Port	Function	Reset State	Pull-up
sim_pin_srst1	out	Reset signal for port1	0	Active
sim_pin_sven1	out	Vcc enable signal for port1	0	Active
sim_pin_data1_tx_out	out	Transmit data signal for port1	0	Active/Passive
pin_sim_rcvd1_in	in	Receive data signal for port1	—	—
pin_sim_simpd1	in	Card insertion detect signal for port1	—	—
Module Interrupts				
ipi_int_sim_ipb_int	out	Active high SIM Interrupt. Composed of OEF, XTE, SDI1 and SDI0.	0	—
ipi_int_sim_data_irq	out	Active high SIM Data Interrupt. Composed of TC, ETC, TFE, and RDRF.	0	—
Module DMA Requests				
ipd_sim_tx_dmareq_b	out	Active low. Transmitter DMA request	1	—
ipd_sim_rx_dmareq_b	out	Active low. Receiver DMA request	1	—
Alternate SIM Signals				
sim_arcvd_in	out	Alternate Receive data line. Output to secondary SIM module	1	—
sim_asimpd	out	Alternate SIM presence detect line. Output to secondary SIM module	0	—
sim_aclk	in	Alternate SIM clock. Input from secondary SIM module	—	—
sim_arst	in	Alternate SIM reset line. Input from secondary SIM module	—	—
sim_adata_tx_out	in	Alternate transmit data line. Input from secondary SIM module	—	—
sim_adata_tx_gz	in	Alternate transmit tri-state control signal. Input from secondary SIM module	—	—
sim_aven	in	Alternate SIM vcc enable line. Input from secondary SIM module	—	—
sim_arcvd_io	out	I/O Alternate Receive data line. Output to secondary SIM module	1	—

51.2.2 Detailed Signal Descriptions

The following subsections discuss the signals.

51.2.2.1 sim_pin_sclk0

The sim_pin_sclk0 is an output from the chip to the SmartCard. This signal is the clock that the SIM module provides for the SmartCard. Typical frequencies are 1 MHz to 5 MHz. This clock is 372 times the data rate that is on pin sim_pin_data0_tx_out. There is no required timing relationship between this clock signal and any of the other data signals. This is because of the asynchronous nature of the protocol.

51.2.2.2 sim_pin_srst0

The sim_pin_srst0 is the reset signal from the SIM to the SmartCard.

51.2.2.3 sim_pin_sven0

The sim_pin_sven0 is the SmartCard power supply enable control signal.

51.2.2.4 sim_pin_data0_tx_out

The sim_pin_data0_tx_out is the data transmitted from the SIM module to the SmartCard. In a 1-wire mode of operating, this output port must be bidirectional with a pull-up resister off chip.

51.2.2.5 pin_sim_rcvd0_in

The pin_sim_rcvd0_in is the receive data coming from the SmartCard to the SIM module.

51.2.2.6 pin_sim_simpd0

The pin_sim_simpd0 is the SmartCard insertion detect.

51.2.2.7 sim_pin_sclk1

The sim_pin_sclk1 is an output from the chip to the smartcard. This signal is the clock that the SIM module provides for the smartcard. Typical frequencies are 1 MHz to 5 MHz. This clock is 372 times the data rate that is on pin sim_pin_data1_tx_out. There is no required timing relationship between this clock signal and any of the other data signals. This is because of the asynchronous nature of the protocol.

51.2.2.8 sim_pin_srst1

The sim_pin_srst1 is the reset signal from the SIM to the SmartCard.

51.2.2.9 sim_pin_sven1

The sim_pin_sven1 is the SmartCard power supply enable control signal.

51.2.2.10 sim_pin_data1_tx_out

The sim_pin_data1_tx_out is the data transmitted from the SIM module to the SmartCard. In a one wire mode of operating, this output port must be bidirectional with a pull-up resister off chip.

51.2.2.11 pin_sim_rcvd1_in

The pin_sim_rcvd1_in is the receive data coming from the SmartCard to the SIM module.

51.2.2.12 pin_sim_simpd1

The pin_sim_simpd1 is the SmartCard insertion detect.

51.2.2.13 ipg_enable_clk

The ipg_enable_clk is to control external ipg_clk gating when needed. SIM module has no dedicated signal to gating the ipg_perclk, but the Clock Controller module can synchronize ipg_enable_clk to ipg_perclk domain to gating ipg_perclk.

51.3 Memory Map and Register Definition

Section 51.3.3, “Register Descriptions” on page 51-XIII provides the detailed descriptions for all of the SIM registers.

51.3.1 Memory Map

Table 51-7 shows the SIM memory map.

Table 51-7. SIM Memory Map

Address	Register	Access	Reset Value	Section/Page
0x5001_8000 (PORT1_CNTL)	SIM Port1 Control register	R/W	0x0000_0000	51.3.3.1/51-XIII
0x5001_8004 (SETUP)	SIM Setup register	R/W	0x0000_0000	51.3.3.2/51-XV
0x5001_8008 (PORT1_DETECT)	SIM Port 1 Detect register	R/W	0x0000_— — — —	51.3.3.3/51-XVI
0x5001_800C (XMT_BUF)	SIM Transmit Buffer register	R/W	0x0000_0000	51.3.3.4/51-XVII
0x5001_8010 (RCV_BUF)	SIM Receive Buffer register	R	0x0000_0000	51.3.3.5/51-XVII
0x5001_8014 (PORT0_CNTL)	SIM Port0 Control register	R/W	0x0000_0000	51.3.3.6/51-XIX
0x5001_8018 (CNTL)	SIM Control register	R/W	0x0000_0006	51.3.3.7/51-XX
0x5001_801C (CLK_PRESCALER)	SIM Clock Prescaler register	R/W	0x0000_0002	51.3.3.8/51-XXII
0x5001_8020(RCV_THRESHOLD)	SIM Receive Threshold register	R/W	0x0000_0001	51.3.3.9/51-XXIII
0x5001_8024 (ENABLE)	SIM Enable register	R/W	0x0000_0000	51.3.3.10/51-XXIV

Table 51-7. SIM Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0x5001_8028 (XMT_STATUS)	SIM Transmit Status register	R/W	0x0000_00B8	51.3.3.11/51-XXV
0x5001_802C (RCV_STATUS)	SIM Receive Status register	R/W	0x0000_0040	51.3.3.12/51-XXVII
0x5001_8030 (INT_MASK)	SIM Interrupt Mask register	R/W	0x0000_1FFF	51.3.3.13/51-XXIX
0x5001_803C (PORT0_DETECT)	SIM Port0 Detect register	R/W	0x0000_— — — —	51.3.3.14/51-XXXI
0x5001_8040 (DATA_FORMAT)	SIM Data Format register	R/W	0x0000_0000	51.3.3.15/51-XXXII
0x5001_8044(XMT_THRESHOLD)	SIM Transmit Threshold register	R/W	0x0000_0000	51.3.3.16/51-XXXIII
0x5001_8048 (GUARD_CNTL)	SIM Transmit Guard Control register	R/W	0x0000_0000	51.3.3.17/51-XXXIV
0x5001_804C (OD_CONFIG)	SIM Open Drain Configuration Control register	R/W	0x0000_0000	51.3.3.18/51-XXXV
0x5001_8050 (RESET_CNTL)	SIM Reset Control register	R/W	0x0000_0000	51.3.3.19/51-XXXVI
0x5001_8054 (CHAR_WAIT)	SIM Character Wait Time register	R/W	0x0000_FFFF	51.3.3.20/51-XXXVII
0x5001_8058 (GPCNT)	SIM General Purpose Counter register	R/W	0x0000_FFFF	51.3.3.21/51-XXXVIII
0x5001_805C (DIVISOR)	SIM Divisor register	R/W	0x0000_00FF	51.3.3.22/51-XXXIX
0x5001_8060 (BWT)	SIM Block Wait Time register	R/W	0x0000_FFFF	51.3.3.23/51-XXXIX
0x5001_8064 (BGT)	SIM Block Guard Time register	R/W	0x0000_0000	51.3.3.24/51-XL
0x5001_8068 (BWT_H)	SIM Block Wait Time register HIGH	R/W	0x0000_FFFF	51.3.3.25/51-XLI
0x5001_806C (XMT_FIFO_STAT)	SIM Transmit FIFO Status register	R	0x0000_0000	51.3.3.26/51-XLII
0x5001_8070 (RCV_FIFO_CNT)	SIM Receive FIFO Counter register	R	0x0000_0000	51.3.3.27/51-XLIII
0x5001_8074 (RCV_FIFO_WPTR)	SIM Receive FIFO Write Pointer register	R	0x0000_0000	51.3.3.28/51-XLIV
0x5001_8078 (RCV_FIFO_RPTR)	SIM Receive FIFO Read Pointer register	R	0x0000_0000	51.3.3.29/51-XLV

51.3.2 Register Summary

Figure 51-2 shows the key to the register fields and Table 51-8 shows the register figure conventions.

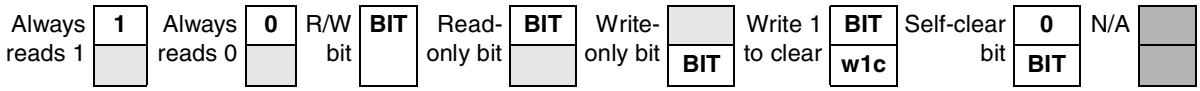


Figure 51-2. Key to Register Fields

Table 51-8. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 51-9 shows SIM register summary.

Table 51-9. SIM Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x5001_8000 (PORT1_CNTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0		3VOLT1	SCSP1	SCEN1	SRST1	STEN1	SVEN1	SAPD1
	W									SFPD1							
0x5001_8004 (SETUP)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SPS	AMODE
	W																

Table 51-9. SIM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x5001_8008 (PORT1_DETECT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	SPDS1	SPDP1	SDI1	SDIM1
	W															w1c	
0x5001_800C (XMT_BUF)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	XMT[7:0]							
	W																
0x5001_8010 (RCV_BUF)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	CWT	FE	PE	RCV[7:0]							
	W																
0x5001_8014 (PORT0_CNTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0		3VOLT0	SCSP0	SCEN0	SRST0	STEN0	SVEN0	SAPD0
	W									SFPD0							
0x5001_8018 (CNTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	BWTEN	XMT_CRC_LRC	CRCEN	LRCEN	CWTEN	GPCNTSEL[1:0]		CLK	BAUD_SEL[2:0]			SAMPLE12	ONACK	ANACK	ICM	0
	W																
0x5001_801C (CLK_PRESCALER)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	CLK_PRESCALER[7:0]							
	W																
0x5001_8020(RCV_THRESHOLD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	RTH[3:0]				RDT[8:0]								
	W																
0x5001_8024 (ENABLE)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	TXDMAEN	RXDMAEN	XMTEN	RCVEN
	W																
0x5001_8028 (XMT_STATUS)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	GPCNT	TDTF	TFO	TC	ETC	TFE	0	0	XTE
	W								w1c	w1c	n/a	w1c	w1c	w1c			w1c
0x5001_802C (RCV_STATUS)	R	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	W																
	R	0	0	0	0	BGT	BWT	RTE	CWT	CRCOK	LRCOK	RDRF	RFD	0	0	RFE	OEF
	W					w1c	w1c	w1c	w1c			w1c				w1c	w1c

Table 51-9. SIM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x5001_8030 (INT_MASK)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
	W																
	R	0	0	RFEM	BGTM	BWTM	RTM	CWTM	GPCNTM	TDTFM	TFOM	XTM	TFEIM	ETCIM	OIM	TCIM	RIM
	W																
	W																
0x5001_803C (PORT0_DETECT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	SPDS0	SPDP0	SDI0	SDIM0
	W															w1c	
0x5001_8040 (DATA_FORMAT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	IC
	W																
0x5001_8044(XMT_THRESHOLD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	XTH[3:0]				TDT3:0]			
	W																
0x5001_8048 (GUARD_CNTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	RCVR11	GETU[7:0]							
	W																
0x5001_804C (OD_CONFIG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OD_P1	OD_P0
	W																
0x5001_8050 (RESET_CNTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	DBUG	STOP	DOZE	KILL CLOCK		FLUSH XMT	FLUSH RCV
	W														SOFT RESET		
0x5001_8054 (CHAR_WAIT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	CHARACTER WAIT TIME[15:0]															
	W																
0x5001_8058 (GPCNT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	GENERAL PURPOSE COUNTER[15:0]															
	W																
0x5001_805C (DIVISOR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	DIVISOR[7:0]							
	W																

Table 51-9. SIM Register Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x5001_8060 (BWT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	BLOCK WAIT TIME[15:0]															
	W																
0x5001_8064 (BGT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	BLOCK GUARD TIME[15:0]															
	W																
0x5001_8068 (BWT_H)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	BLOCK WAIT TIME HIGH[15:0]															
	W																
0x5001_806C (XMT_FIFO_STAT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	XMT_CNT[3:0]			XMT_WPTR[3:0]					XMT_RPTR[3:0]			
	W																
0x5001_8070 (RCV_FIFO_CNT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	RCV_CNT[8:0]								
	W																
0x5001_8074 (RCV_FIFO_WPTR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	RCV_WPTR[8:0]								
	W																
0x5001_8078 (RCV_FIFO_RPTR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	RCV_RPTR[8:0]								
	W																

51.3.3 Register Descriptions

This section contains the detailed descriptions for the SIM registers.

51.3.3.1 SIM Port1 Control Register (PORT1_CNTL)

See Figure 51-3 for illustration of valid bits in the SIM Port1 Control Register and Table 51-10 for description of the bit fields.

0x5001_8000 (PORT1_CNTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	3VOLT1	SCSP1	SCEN1	SRST1	STEN1	SVEN1	SAPD1
W									SFPD1							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-3. SIM Port1 Control Register

Table 51-10. SIM Port1 Control Register Field Descriptions

Field	Description
31–8	Reserved
7 SFPD1	Auto Power Down port1. Writing a “1” to this location will start the auto power down sequence for port 1. This bit will autoclear. A read of this bit will give a “0”. 0 No effect 1 Start Auto Power down
6 3VOLT1	External one wire interface for SIM Card port1. Used to configure the Port 1 transmit pin as bi-directional. This allows the Port 1 Receive pin to be re-used as General Purpose I/O. This operation is restricted to bi-directional data SIM cards only. This bit should not be changed while the receiver or transmitter is enabled! 0 Port1 uses both RCV and XMT pins 1 Port1 XMT pin bidirectional. Port1 RCV PIN unused
5 SCSP1	SIM Card Clock Stop Polarity Port1. Used to control the polarity of the idle SIM clock when the clock is disabled by SCEN1. It will be forced low by hardware during the auto power down sequence. This forces the clock be a logic 0 when stopped by auto power down as required by ISO 7816 spec. 0 Clock is logic 0 when stopped by SCEN1 1 Clock is logic 1 when stopped by SCEN1
4 SCEN1	SIM card Clock Enable Port 1. Used to enable/disable the clock to the SIM card. It can be forced low by hardware during the auto power down sequence. 0 SIM Card Clock Enabled Port1 1 SIM Card Clock Disabled Port1
3 SRST1	SIM card Reset. Used to control state of reset line to the SIM card. It can be forced low by hardware during the auto power down sequence. SIM card reset signals are active low. 0 SIM Card reset Port1 inactive (default) 1 SIM Card reset Port1 active
2 STEN1	SIM card Transmit Enable Port 1. Used to enable/disable the XMT data to the SIM card. It can be forced low by hardware during the auto power down sequence. 0 Port1 Transmit Data is forced to zero (default) 1 Port 1 Transmit Data controlled by SIM module

Table 51-10. SIM Port1 Control Register Field Descriptions (continued)

Field	Description
1 SVEN1	SIM card Vcc Enable Port 1. Used to control the state of the SVEN1 pin on SIM card port 1. The SVEN1 pin controls the SIM card Vcc enable in the power management chip. It can be forced low by hardware during the auto power down sequence. 0 SIM card Voltage Port 1 disabled (default) 1 SIM card Voltage Port 1 enabled
0 SAPD1	SIM card Auto Power Down Port 1. Used to enable/disable the auto power down function for port 1. It will be forced low at the end of the auto power down sequence. 0 Auto power down Port 1 disabled (default) 1 Auto power down Port 1 enabled

51.3.3.2 SIM Setup Register (SETUP)

See Figure 51-4 for illustration of valid bits in the SIM Setup Register and Table 51-11 for description of the bit fields.

0x5001_8004 (SETUP) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SPS	AMODE
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-4. SIM Setup Register

Table 51-11. SIM Setup Register Field Descriptions

Field	Description
31–2	Reserved
1 SPS	SIM card Port Select. Controls which port the SIM interface uses. **Note:** The AMODE bit must be zero when the SPS bit is set to 1. 0 Port 0 Enabled (default) 1 Port 1 Enabled
0 AMODE	Alternate SIM Card Mode enable. Enables an alternate SIM module to control SIM card Port 1. **Note:** The SPS bit must be 0 to give the alternate SIM module control. 0 Alternate Port Disabled (default) 1 Alternate Port Enabled

51.3.3.3 SIM Port1 Detect Register (PORT1_DETECT)

See Figure 51-5 for illustration of valid bits in the SIM Port1 Detect Register and Table 51-12 for a detailed description of the bit fields. A summary is as follows:

SPDS1 determines which edge transition of the SIMPD1 pin is used for SIM card presence detection. Presence detection can be used to determine if the card has been inserted or removed. The occurrence of the SIMPD1 edge specified by SPDS1 causes the following:

- SDI1 is set.
- If the SDIM1 mask is clear, an interrupt on SIMIRQ_N
- If SAPD1 in the PORT1_CNTL register is set, an auto power down sequence begins.

If SIM card insertion is expected, SAPD1 can be set low to avoid the auto-power-down sequence. There is no auto-power-up sequence. The bit SPDP1 can be used to determine the current state of the SIMPD1 pin.

0x5001_8008 (PORT1_DETECT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	SPDS1	SPDP1	SDI1	SDIM1
W															w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	—	0	1

Figure 51-5. SIM Port 1 Detect Register

Table 51-12. SIM Port 1 Detect Register Field Descriptions

Field	Description
31–4	Reserved
3 SPDS1	SIM Presence Detect Select Port 1. Controls which edge of the SIMPD1 pin is used to detect the presence of the SIM card. 0 Falling edge of SIMPD1 Input (default) 1 Rising edge of SIMPD1 Input
2 SPDP1	SIMPD1 input pin status. This bit reflects the state of the SIMPD1 pin. It is not a latched register bit, but instead a synchronized version of the state of the SIMPD1 pin itself. 0 SIMPD1 pin is logic low 1 SIMPD1 pin is logic high

Table 51-12. SIM Port 1 Detect Register Field Descriptions (continued)

Field	Description
1 SDI1	SIM Detect Interrupt Flag Port 1. Status flag to indicate the insertion or removal of a SIM card has been detected on port 1. Can create an interrupt to the MCU if SDIM1 is low. Write a “1” to this bit to clear. 0 No insertion or removal of SIM card detected on Port 1(default) 1 Insertion or removal of SIM card detected on Port 1
0 SDIM1	SIM Detect Interrupt Mask Port 1. Interrupt mask for the SDI1 interrupt flag. 0 SDI1 enabled 1 SDI1 masked (default)

51.3.3.4 SIM Transmit Buffer Register (XMT_BUF)

See Figure 51-6 for illustration of valid bits in the SIM Transmit Buffer Register and Table 51-13 for description of the bit fields.

0x5001_800C (XMT_BUF) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	XMT[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-6. SIMTransmit Buffer Register

Table 51-13. SIMTransmit Buffer Register Field Descriptions

Field	Description
31–8	Reserved
7–0 XMT	Transmit Buffer. Write to the next available location in the transmit buffer. Writes to this register are ignored when the SPS bit is zero.

51.3.3.5 SIM Receive Buffer Register (RCV_BUF)

See Figure 51-7 for illustration of valid bits in the SIM Receive Buffer Register and Table 51-14 for description of the bit fields.

0x5001_8010 (RCV_BUF) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CWT	FE	PE	RCV[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-7. SIM Port 1 Receive Buffer Register

Table 51-14. SIM Port 1 Receive Buffer Register Field Descriptions

Field	Description
31–11	Reserved
10 CWT	CWT flag. The CWT indicates that this byte was late. It is not necessary to clear the byte since it is overwritten by the next byte received into that location of the FIFO. 0 Byte was on time 1 Byte was late
9 FE	Frame Error flag. The FE flag indicates whether a frame error was detected during the reception of the corresponding byte read in the RCV field. The FE flag cannot create an interrupt. It need not be cleared since it will be overwritten by the next byte received into that location of the FIFO. 0 Byte contains no framing error (default) 1 Byte contains a framing error
8 PE	Parity Error flag. The FE flag indicates whether a parity error was detected during the reception of the corresponding byte read in the RCV field. The PE flag cannot create an interrupt. It need not be cleared since it will be overwritten by the next byte received into that location of the FIFO. A parity error can create a NACK pulse. 0 Byte contains no parity error (default) 1 Byte contains a parity error
7–0 RCV	Receive buffer. Read from the next location in the receive buffer. Reads from this register return zero when the SPS bit is zero.

51.3.3.6 SIM Port0 Control Register (PORT0_CNTL)

See Figure 51-8 for illustration of valid bits in the SIM Port0 Control Register and Table 51-15 for description of the bit fields.

0x5001_8014 (PORT0_CNTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0		3VOLT0	SCSP0	SCEN0	SRST0	STEN0	SVEN0	SAPD0
W									SFPD0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-8. SIM Port0 Control Register

Table 51-15. SIM Port0 Control Register Field Descriptions

Field	Description
31–8	Reserved
7 SFPD0	Auto Power Down port0. Writing a “1” to this location will start the auto power down sequence for port 0. This bit will autoclear.A read of this bit will give a “0”. 0 No effect 1 Start Auto Power down
6 3VOLT0	External one wire interface for SIM Card port0. Used to configure the Port 0 transmit pin as bi-directional. This allows the Port 0 Receive pin to be re-used as General Purpose I/O. This operation is restricted to bi-directional data SIM cards only. This bit should not be changed while the receiver or transmitter is enabled! 0 Port0 uses both RCV and XMT pins 1 Port0 XMT pin bidirectional. Port0 RCV pin unused
5 SCSP0	SIM Card Clock Stop Polarity port0. Used to control the polarity of the idle SIM clock when the clock is disabled by SCEN0. It will be forced low by hardware during the auto power down sequence. This forces the clock to be a logic 0 when stopped by auto power down as required by ISO 7816 spec. 0 Clock is logic 0 when stopped by SCEN0 1 Clock is logic 1 when stopped by SCEN0
4 SCEN0	SIM card Clock Enable Port 0. Used to enable/disable the clock to the SIM card. It can be forced low by hardware during the auto power down sequence. 0 SIM Card Clock Disabled Port 0 1 SIM Card Clock Enabled Port 0
3 SRST0	SIM card Reset. Used to control state of reset line to the SIM card. It can be forced low by hardware during the auto power down sequence. SIM card reset signals are active low. 0 SIM Card reset Port0 inactive (default) 1 SIM Card reset Port0 active

Table 51-15. SIM Port0 Control Register Field Descriptions (continued)

Field	Description
2 STEN0	SIM card Transmit Enable Port 0. Used to enable/disable the XMT data to the SIM card. It can be forced low by hardware during the auto power down sequence. 0 Port0 Transmit Data is forced to zero (default) 1 Port 0 Transmit Data controlled by SIM module
1 SVEN0	SIM card Vcc Enable Port 0. Used to control the state of the SVEN0 pin on SIM card port 0. The SVEN0 pin controls the SIM card Vcc enable in the power management chip. It can be forced low by hardware during the auto power down sequence. 0 SIM card Voltage Port 0 disabled (default) 1 SIM card Voltage Port 0 enabled
0 SAPD0	SIM card Auto Power Down Port 0. Used to enable/disable the auto power down function for port 0. It will be forced low at the end of the auto power down sequence. 0 Auto power down Port 0 disabled (default) 1 Auto power down Port 0 enabled

51.3.3.7 SIM Control Register (CNTL)

See Figure 51-9 for illustration of valid bits in the SIM Control Register and Table 51-16 for description of the bit fields.

0x5001_8018 (CNTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWTEN	XMT_CRC_LRC	CRCEN	LRCEN	CWTEN	GPCNT_CLK_SEL[1:0]			BAUD_SEL[2:0]		0	SAMPLE12	ONACK	ANACK	ICM	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Figure 51-9. SIM Control Register

Table 51-16. SIM Control Register Field Descriptions

Field	Description
31–16	Reserved
15 BWTEN	Block wait time enable. Writing a “1” to this bit will enable the BWT and BGT functions. The BWT and BGT functions can then be individually selected using the interrupt mask. 0 Disable BWT, BGT 1 Enable BWT, BGT

Table 51-16. SIM Control Register Field Descriptions (continued)

Field	Description
14 XMT_CRC_LRC	Transmit CRC or LRC. This bit specifies whether or not to transmit the redundancy checking data at the end of a transmission (that is, when the FIFO becomes empty). 0 No Redundancy check info transmitted (default) 1 Transmit LRC or CRC info when FIFO empties (whichever is enabled)
13 CRCEN	CRC Enable. This bit enables the calculation of the 16-bit CRC value for both receiver and transmitter. The result of the calculation is continuously compared to the expected remainder and reflected in the CRCOK bit in the RCV_STAT register. Clearing this bit resets the current CRC residual value in the SIM hardware. 0 16-bit Cyclic Redundancy Checking disabled (default) 1 16-bit Cyclic Redundancy Checking enabled
12 LRCEN	LRC Enable. This bit enables the calculation of the 8-bit LRC value for both receiver and transmitter. The result of the calculation is continuously compared to zero and reflected in the LRCOK bit in the RCV_STAT register. Clearing this bit resets the current LRC value in the SIM hardware. 0 8-bit Linear Redundancy Checking disabled (default) 1 8-bit Linear Redundancy Checking enabled
11 CWTEN	Character Wait Time Counter Enable. Enables the character wait time counter. Clearing this bit resets the counter to zero. 0 Character Wait time Counter off (default) 1 Character Wait time counter on
10–9 GPCNT_CLK_SEL[1:0]	General Purpose Counter Clock Select. Selects which clock source is used by SIM Module general purpose counter. The only way to reset the counter is to set these bits to zero. The counter will begin counting as soon as the clock input is selected and the clocks are enabled. These input clocks are enabled through other register bits of the SIM module (KILL_CLOCK, RCV_EN, and XMT_EN respectively). 00 Disabled / Reset 01 Card Clock 10 Receive Clock 11 ETU Clock (transmit clock)
8–6 BAUD_SEL[2:0]	SIM Baud Rate Select. Selects the asynchronous baud rate divisor of the clock When set to “111”, the divisor is set to the value programmed in the DIVISOR register. This allows for more flexible baud rate determination. 000 31 (372/1 Fi/Di) 001 32 (512/2 Fi/Di) 010 16 (512/4 Fi/Di) 011 8 (512/8 Fi/Di) 100 4 (512/16 Fi/Di) 101 2 (512/32 Fi/Di) 110 1 (512/64 Fi/Di) 111 DIVISOR Reg
5	Reserved
4 SAMPLE12	Sample12. Set the third stage divider. This sets the corresponding sample rate which is the number of times a bit being received is sampled. 0 divide by 8(default) 1 divide by 12
3 ONACK	Overrun NACK Enable. Enables overrun NACK generation. 0 NACK generation on overrun is disabled (default) 1 NACK generation on overrun is enabled

Table 51-16. SIM Control Register Field Descriptions (continued)

Field	Description
2 ANACK	Automatic NACK Enable. Enables nack generation for parity errors or invalid initial characters when in ICM mode. 0 NACK generation on errors disabled 1 NACK generation on errors enabled (default)
1 ICM	Initial Character Mode. Enables initial character mode. Will be automatically cleared by hardware once a valid initial character is received. 0 Initial Character Mode disabled 1 Initial Character Mode enabled (default)
0	Reserved

51.3.3.8 SIM Clock Prescaler Register (CLK_PRESCALER)

See Figure 51-10 for illustration of valid bits in the SIM Clock Prescaler Register and Table 51-17 for description of the bit fields.

0x5001_801C (CLK_PRESCALER) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	CLK_PRESCALER[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 51-10. SIM Clock Prescaler Register

Table 51-17. SIM Clock Prescaler Register Field Descriptions

Field	Description
31–8	Reserved
7–0 CLK_PRESCALER	Clock prescaler divisor register. The value written to this register will determine the first stage divider setting. If the ipg_perclk is 66Mhz, a typical setting would be 0x0e. This would set the SIM card clock to 66Mhz/14 = 4.7Mhz. The duty cycle of divided clock will be between 45% and 55% according to ISO7816 requirement. For the odd divider factor (2K+1), the duty cycle will be K/(2K+1) and (K+1)/(2k+1). So for 0x03, the duty cycle is 33%-66%. For 0x05, the duty cycle is 40%-60%. For 0x07, the duty cycle is 43%-57%, and for 0x09, the duty cycle is 44%-56%. For all other values, the clock duty cycle can meet the ISO7816 requirement. 0x00~0x02 ipg_perclk/ 2 0x03 ipg_perclk / 3 0x04 ipg_perclk / 4 0x05 ipg_perclk / 5 0x06 ipg_perclk / 6 ... 0xFD ipg_perclk / 253 0xFE ipg_perclk / 254 0xFF ipg_perclk / 255

51.3.3.9 SIM Receive Threshold Register (RCV_THRESHOLD)

See Figure 51-11 for illustration of valid bits in the SIM Receive Threshold Register and Table 51-18 for description of the bit fields.

0x5001_8020(RCV_THRESHOLD) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	RTH[3:0]					RDT[8:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 51-11. SIM Receive Threshold Register

Table 51-18. SIM Receive Threshold Register Field Descriptions

Field	Description
31–13	Reserved

Table 51-18. SIM Receive Threshold Register Field Descriptions (continued)

Field	Description
12–9 RTH[3:0]	Receive Nack Threshold. Used to specify the number of consecutive NACK's transmitted by the SIM module, for a given character, before the receive threshold error (RTE) flag is triggered. A value of 0 indicates that RTE is never set. When a valid character is received by the SIM, the internal counter keeping track of the NACK count resets to zero for the subsequent byte being received. If the ANACK bit is clear in the CNTL register, RTH has no effect.
8–0 RDT[8:0]	Receive Data Threshold. Determines the number of unread bytes that must exist in the FIFO to trigger the receive data register full (RDRF) interrupt flag. If the number of unread bytes in the receive FIFO is greater than or equal to the value in RDT, the RDRF flag in the RCV_STATUS register will be set. A value of zero indicates that there must be 288 unread bytes in the FIFO to trigger RDRF. The RDT value can be altered at any time, and the RDRF flag will be updated accordingly. If a value of more than 288 is entered in this register, the register value will remain 288.

51.3.3.10 SIM Enable Register (ENABLE)

See Figure 51-12 for illustration of valid bits in the SIM Enable Register and Table 51-19 for description of the bit fields.

0x5001_8024 (ENABLE) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	TXDMAEN	RXDMAEN	XMTEN	RCVEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-12. SIM Enable Register

Table 51-19. SIM Enable Register Field Descriptions

Field	Description
31–2	Reserved
3 TXDMA_EN	Transmitter DMA Enable. This bit allows SIM to request for DMA transfers. When enabled, DMA requests are generated when the number of bytes in the transmit FIFO is less than or equal to the value programmed in the TDT[3:0] bits in the XMT_THRESHOLD register and XMT_EN is set. 0 SIM Transmitter DMA requests disabled. 1 SIM Transmitter DMA requests enabled.
2 RXDMA_EN	Receiver DMA Enable. This bit allows SIM to request for DMA transfers. When enabled, DMA requests are generated when the number of bytes in the receive FIFO is more than or equal to the value programed in the RDT[8:0] in the RCV_THRESHOLD register and RCV_EN is set 0 SIM Receiver DMA request disabled. 1 SIM Receiver DMA request enabled.

Table 51-19. SIM Enable Register Field Descriptions (continued)

Field	Description
1 XMT_EN	SIM Transmit Enable. Used to enable/disable the SIM transmit state machine. When the SIM is being used to receive data, XMT_EN should be deasserted. This bit also enables the xmt_clk and rcv_clk inputs to the General Purpose Counter. **Note:** Setting this bit (transition from 0 to 1) will reset the CRC and LRC values. 0 SIM Transmitter disabled (default) 1 SIM Transmitter enabled
0 RCV_EN	SIM Receiver Enable. Used to enable/disable the SIM receive state machine. The RCV_EN bit should be left high whenever the SIM module is in use. The SIM module has an automatic receive mode operation that disables the reception of characters when the transmitter is operational. Once the transmitter has completed sending the last character, the receiver is automatically enabled. This bit also enables the RCV_CLK input to the General Purpose Counter. 0 SIM Receiver disabled (default) 1 SIM Receiver enabled

51.3.3.11 SIM Transmit Status Register (XMT_STATUS)

See Figure 51-13 for illustration of valid bits in the SIM Transmit Status Register and Table 51-20 for description of the bit fields.

0x5001_8028 (XMT_STATUS) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	GPCNT	TDTF	TFO	TC	ETC	TFE	0	0	XTE
W								w1c	w1c	n/a	w1c	w1c	w1c			w1c
Reset	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0

Figure 51-13. SIM Transmit Status Register

Table 51-20. SIM Transmit Status Register Field Descriptions

Field	Description
31–9	Reserved
8 GPCNT	General purpose Counter Flag. Used to indicate when the General purpose counter has reached the value in the GPCNT register. 0 GPCNT time not reached, or bit has been cleared. (default). 1 General Purpose counter has reached the GPCNT value.

Table 51-20. SIM Transmit Status Register Field Descriptions (continued)

Field	Description
7 TDTF	Transmit Data Threshold Flag. Used to indicate when the number of bytes in the transmit FIFO is less than or equal to the value programmed in the TDT[3:0] bits in the XMT_THRESHOLD register. 0 Number of bytes in FIFO is greater than TDT[3:0], or bit has been cleared. 1 Number of bytes in FIFO is less than or equal to TDT[3:0] (default)
6 TFO	Transmit FIFO Overfill Error. Used to indicate when the Transmit FIFO has been written with more than 16 bytes. The TFO bit can only be cleared by setting the FLUSH_XMT or SOFT_RESET bit in the RESET_CNTL register. 0 No transmit FIFO overfill error has occurred (default). 1 A Transmit FIFO overfill error has occurred.
5 TC	Transmit Complete. Used to indicate whether the SIM transmitter is ready for a new transmission. The TC flag becomes set after the guard time has expired for the last byte in the transmit FIFO. The TC flag will create an interrupt if TCIM in the INT_MASK register is low. The TC bit is a write-one-to-clear bit. 0 Transmit pending or in progress 1 Transmit complete (default)
4 ETC	Early Transmit Complete. Used to indicate that the SIM transmitter has finished sending the current byte and the transmit FIFO is empty. This bit differs from the TC bit in that it is set before the guard time of the last byte has elapsed. The ETC flag will create an interrupt if ETCIM in the INT_MASK register is low. The ETC bit is a write-one-to-clear bit. 0 Transmit pending or in progress 1 Transmit complete (default)
3 TFE	Transmit FIFO Empty. Used to indicate that the SIM transmit FIFO has emptied. This bit will be set when the last byte in the transmit FIFO has been transferred to the SIM transmitter shift register. The TFE flag will create an interrupt if TFEIM in the INT_MASK register is low. The TFE bit is a write-one-to-clear bit. 0 Transmit FIFO is not empty 1 Transmit FIFO is empty (default)
2–1	Reserved
0 XTE	Transmit NACK Threshold Error. Used to indicate the transmit NACK threshold has been reached. When XTE is high, no further transmissions will be done until the XTE flag is cleared. Any data transmissions still pending in the transmit FIFO will be aborted, and the TC, ETC, and TFE flags will be set. The XTE flag will create an interrupt if XTM in the IMT_MASK register is low. The XTE bit is a write-one-to-clear bit. 0 Transmit NACK threshold has not been reached (default) 1 Transmit NACK threshold reached; transmitter frozen

51.3.3.12 SIM Receive Status Register (RCV_STATUS)

See Figure 51-14 for illustration of valid bits in the SIM Receive Status Register and Table 51-21 for description of the bit fields.

0x5001_802C (RCV_STATUS) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	BGT	BWT	RTE	CWT	CRC OK	LRCO K	RDRF	RFD	0	0	RFE	OEF
W					w1c	w1c	w1c	w1c			w1c				w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

Figure 51-14. SIM Receive Status Register

Table 51-21. SIM Receive Status Register Field Descriptions

Field	Description
31–12	Reserved
11 BGT	Block guard time error flag. Used to indicate if the block guard time was too small. The threshold is set by the block guard time register. 0 Block guard time was sufficient. 1 Block guard time was too small.
10 BWT	Block wait time error flag. Used to indicate if the block wait time has been exceeded. The threshold is set by the block wait time registers. 0 Block wait time not exceeded. 1 Block wait time was exceeded.
9 RTE	Receive NACK threshold error flag. Used to indicate whether the number of consecutive NACK's generated by the SIM module in response to receive parity errors, for the byte being received, equals the value programmed in the RTH[3:0] in the RCV_THRESHOLD register. This bit would never be set unless the ANACK bit is set in the CNTL register. The SAPDx bit in the PORTx_CNTL resister must be set to enable the threshold error to trigger the auto power down sequence. RTE is a write once to clear bit. Clearing this bit also resets the internal counter for consecutive NACK's being transmitted for a given byte. 0 Number of NACKs generated by the receiver is less than the value programmed in RTH[3:0] 1 Number of NACKs generated by the receiver is equal to the value programmed in RTH[3:0]
8 CWT	Character Wait Time Counter Flag. Used to indicate when the time between received characters is equal to or greater than the value programmed in the CHAR_WAIT register. 0 No CWT violation has occurred (default). 1 Time between two consecutive characters exceeded the value in CHAR_WAIT.

Table 51-21. SIM Receive Status Register Field Descriptions (continued)

Field	Description
7 CRCOK	Cyclic Redundancy Check Okay flag. Used to indicate when the calculated 16-bit CRC value matches the expected value for the current input data stream. The value is calculated across all received characters from the point the CRCEN bit is set in the CNTL register. The current CRC residual can be reset by three mechanisms: • Clear CRCEN bit in CNTL register • Set XMT_EN bit in ENABLE register • Automatically by hardware when ETC flag is set at the end of a transmission. 0 Current CRC value does not match remainder. 1 Current calculated CRC value matches the expected result.
6 LRCOK	Linear Redundancy Check Okay flag. Used to indicate when the calculated 8-bit LRC value is zero value for the current input data stream. The value is calculated across all received characters from the point the LRCEN bit is set in the CNTL register. The current LRC residual can be reset by three mechanisms: • Clear LRCEN bit in CNTL register • Set XMT_EN bit in ENABLE register • Automatically by hardware when ETC flag is set at the end of a transmission. 0 Current LRC value does not match remainder. 1 Current calculated LRC value matches the expected result (that is, zero).
5 RDRF	Receive Data Register Full. Used to indicate when the number of bytes in the receive FIFO is more than or equal to the value programed in the RDT[8:0] in the RCV_THRESHOLD register. 0 Number of unread bytes in receive buffer < value set by RDT[8:0] (default), or bit has been cleared. 1 Number of unread bytes in receive buffer >= value set by RDT[8:0].
4 RFD	Receive FIFO has unread Data. Used to indicate that there is at least one unread byte in the receive data FIFO. Can only be cleared by reading all bytes out of the receive FIFO. The RFD bit cannot be used to create an interrupt. Normally, the SIM triggers the interrupt with RDRF and software uses RFD to read all of the bytes out of the receive FIFO. 0 There are no unread bytes in the receive FIFO (default). 1 There is at least one unread byte in the receiver FIFO.
3–2	Reserved
1 RFE	Receive FIFO Empty. Used to indicate that the SIM receive FIFO has emptied. This bit will be set when the last byte in the receive FIFO has been transferred to the IPS bus. The RFE flag will create an interrupt if RFEIM in the INT_MASK register is low. The RFE bit is a write-one-to-clear bit. 0 Receive FIFO is not empty 1 Receive FIFO is empty (default)
0 OEF	Overrun Error Flag. Used to indicate that the SIM was unable to store received data due to already having 288 unread bytes in the FIFO. It does not necessarily indicate that data has been lost. If the ONACK control bit in the CNTL register is set, there will be a NACK pulse generated on bytes that would otherwise cause a loss of data due to a full FIFO. These bytes should be retransmitted by the SIM card which implies that no data has actually been lost. In this case, the OEF flag is just an indicator that this situation has occurred which may be helpful in system debug. For the case where ONACK is not set, a set OEF flag does indicate a loss of data since all bytes received with the OEF flag set will indeed be lost (including the byte that caused the bit to be set). The OEF flag will cause an interrupt if the OIM bit in the INT_MASK register. The OEF flag is a write-one-to-clear bit. 0 No overrun error has occurred (default). 1 A byte was received when the received FIFO was already full.

51.3.3.13 SIM Interrupt Mask Register (INT_MASK)

See Figure 51-15 for illustration of valid bits in the SIM Interrupt Mask Register and Table 51-22 for description of the bit fields.

0x5001_8030 (INT_MASK) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	RFEM	BGTM	BWTM	RTM	CWTM	GPCNTM	TDTFM	TFOM	XTM	TFEIM	ETCIM	OIM	TCIM	RIM
W																
Reset	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 51-15. SIM Interrupt Mask Register

Table 51-22. SIM Interrupt Mask Register Field Descriptions

Field	Description
31–14	Reserved
13 RFEM	Receive fifo empty interrupt mask. Used to enable/disable the ability of the RFE flag in the RCV_STATUS register to generate SIM interrupts. 0 RFE interrupt enabled 1 RFE interrupt masked (default)
12 BGTM	Block guard time interrupt mask. Used to enable/disable the ability of the BGT flag in the RCV_STATUS register to generate SIM interrupts. 0 BGT interrupt enabled 1 BGT interrupt masked (default)
11 BWTM	Block wait time interrupt mask. Used to enable/disable the ability of the BWT flag in the RCV_STATUS register to generate SIM interrupts. 0 BWT interrupt enabled 1 BWT interrupt masked (default)
10 RTM	Receive Nack threshold interrupt mask. Used to enable/disable the ability of the RTE flag in the RCV_STATUS register to generate SIM interrupts. 0 RTE interrupt enabled 1 RTE interrupt masked (default)
9 CWTM	Character Wait Time Interrupt Mask. Used to enable/disable the ability of the CWT flag in the RCV_STATUS register to generate SIM interrupts. 0 CWT interrupt enabled 1 CWT interrupt masked (default)
8 GPCNTM	General Purpose Counter Interrupt Mask. Used to enable/disable the ability of the GPCNT flag in the XMT_STATUS register to generate SIM interrupts. 0 GPCNT interrupt enabled 1 GPCNT interrupt masked (default)

Table 51-22. SIM Interrupt Mask Register Field Descriptions (continued)

Field	Description
7 TDTFM	Transmit Data Threshold Interrupt Mask. Used to enable/disable the ability of the TDTF flag in the XMT_STATUS register to generate SIM interrupts. 0 TDTF interrupt enabled 1 TDTF interrupt masked (default)
6 TFOM	Transmit FIFO Overfill Error Interrupt Mask. Used to enable/disable the ability of the TFO flag in the XMT_STATUS register to generate SIM interrupts. 0 TFO interrupt enabled 1 TFO interrupt masked (default)
5 XTM	Transmit Threshold Interrupt Mask. Used to enable/disable the ability of the XTE flag in the XMT_STATUS register to generate SIM interrupts 0 XTE interrupt enabled 1 XTE interrupt masked (default)
4 TFEIM	Transmit FIFO Empty Interrupt Mask. Used to enable/disable the ability of the TFE flag in the XMT_STATUS register to generate SIM interrupts. 0 TFE interrupt enabled 1 TFE interrupt masked (default)
3 ETCIM	Early Transmit Complete Interrupt Mask. Used to enable/disable the ability of the ETC flag in the XMT_STATUS register to generate SIM interrupts. 0 ETC interrupt enabled 1 ETC interrupt masked (default)
2 OIM	Overrun Interrupt Mask. Used to enable/disable the ability of the OEF flag in the RCV_STATUS register to generate SIM interrupts. 0 OEF interrupt enabled 1 OEF interrupt masked (default)
1 TCIM	Transmit Complete Interrupt Mask. Used to enable/disable the ability of the TC flag in the XMT_STATUS register to generate SIM interrupts. 0 TC interrupt enabled 1 TC interrupt masked (default)
0 RIM	Receive Interrupt Mask. Used to enable/disable the ability of the RDRF flag in the RCV_STATUS register to generate SIM interrupts. 0 RDRF interrupt enabled 1 RDRF interrupt masked (default)

51.3.3.14 SIM Port0 Detect Register (PORT0_DETECT)

See Figure 51-16 for illustration of valid bits in the SIM Port0 Detect Register and Table 51-23 for description of the bit fields.

0x5001_803C (PORT0_DETECT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	SPDS0	SPDP0	SDI0	SDIM0
W															w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	—	0	1

Figure 51-16. SIM Port0 Detect Register

Table 51-23. SIM Port0 Detect Register Field Descriptions

Field	Description
31–4	Reserved
3 SPDS0	SIM Presence Detect Select Port 0. Controls which edge of the SIMPD0 pin is used to detect the presence of the SIM card. 0 Falling edge of SIMPD0 Input (default) 1 Rising edge of SIMPD0 Input
2 SPDP0	SIMPD0 input pin status. This bit reflects the state of the SIMPD0 pin. It is not a latched register bit, but instead a synchronized version of the state of the SIMPD0 pin itself. 0 SIMPD0 pin is logic low 1 SIMPD0 pin is logic high
1 SDI0	SIM Detect Interrupt flag Port 0. Status flag to indicate the insertion or removal of a SIM card has been detected on port 0. Can create an interrupt to the MCU if SDIM0 is low. Write a “1” to this bit to clear. 0 No insertion or removal of SIM card detected on Port 0 (default) 1 Insertion or removal of SIM card detected on Port 0
0 SDIM0	SIM Detect Interrupt Mask Port 0. Interrupt mask for the SDI0 interrupt flag. 0 SDI0 enabled 1 SDI0 masked (default)

51.3.3.15 SIM Data Format Register (DATA_FORMAT)

See Figure 51-17 for illustration of valid bits in the SIM Data Format Register and Table 51-24 for description of the bit fields.

0x5001_8040 (DATA_FORMAT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	IC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-17. SIM Data Format Register

Table 51-24. SIM Data Format Register Field Descriptions

Field	Description
31–1	Reserved
0 IC	Inverse Convention. Used to configure the SIM to use either inverse convention or direct convention for its data format. The IC bit can be controlled by software, but it is normally set by hardware as a result of the interpretation of the initial character when in ICM mode. 0 Direction convention transfers enabled (default). 1 Inverse convention transfers enabled.

51.3.3.16 SIM Transmit Threshold Register (XMT_THRESHOLD)

See Figure 51-18 for illustration of valid bits in the SIM Transmit Threshold Register and Table 51-25 for description of the bit fields.

0x5001_8044(XMT_THRESHOLD) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	XTH[3:0]				TDT[3:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-18. SIM Transmit Threshold Register

Table 51-25. SIM Transmit Threshold Register Field Descriptions

Field	Description
31–8	Reserved
7–4 XTH[3:0]	Transmit NACK Threshold. Used to set the NACK threshold for the transmitter. Once the threshold number set by XTH has been reached for the current byte being transmitted, the error flag XTE in the XMT_STATUS register will be set. Setting of XTE causes the remaining transmissions queued in the transmit FIFO to be aborted and no more transmissions to occur until software clears XTE. To trigger XTE, a given byte being transmitted must reach the XTH threshold itself. Transmit NACKs accumulated on one byte are not carried over to the next. 0x0XTE will never be set; retransmission after NACK reception is disabled. 0x1XTE will be set after 1 nack is received; 0 retransmissions occurs. 0x2XTE will be set after 2 nacks are received; at most 1 retransmission occurs. 0x3XTE will be set after 3 nacks are received; at most 2 retransmissions occurs. ... ... 0xXXTE will be set after X nacks are received; at most (X - 1) retransmissions occurs. ... ... 0xFXTE will be set after 15 nacks are received; at most 14 retransmissions occurs.
3–0 TDT	Transmit Data Threshold. Used to set the threshold value for the Transmit FIFO at which the TDTF bit in the XMT_STATUS register will be set. When the number of bytes in the Transmit FIFO is less than or equal to TDT[3:0], TDTF will be set.

51.3.3.17 SIM Transmit Guard Control Register (GUARD_CNTL)

See Figure 51-19 for illustration of valid bits in the SIM Transmit Guard Control Register and Table 51-26 for description of the bit fields.

0x5001_8048 (GUARD_CNTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	RCV R11	GETU[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-19. SIM Transmit Guard Control Register

Table 51-26. SIM Transmit Guard Control Register Field Descriptions

Field	Description
31–9	Reserved
8 RCVR11	Receiver use 11 ETUs. Used to configure the SIM module receiver for 11 ETU operation (that is, 1 Stop bit). This bit is provided for support of T=1 cards. 0 Receiver configured for 12 ETU operation (default) 1 Receiver configured for 11 ETU operation
7–0 GETU	Transmit Guard ETUs. Used to control the number of additional Elementary Time Units (ETUs) inserted between bytes transmitted by the SIM transmitter. An ETU is equivalent to one bit time at the given baud rate (for example, the length of a START bit). The guard time has no effect on the SIM receiver. A value of 0x00 inserts no additional ETUs, while a value of 0xFE inserts 254 additional ETUs. A value of 0xFF subtracts one ETU by reducing the number of STOP bits from two to one.

51.3.3.18 SIM Open Drain Configuration Control Register (OD_CONFIG)

See Figure 51-20 for illustration of valid bits in the SIM Open Drain Configuration Control Register and Table 51-27 for description of the bit fields.

0x5001_804C (OD_CONFIG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	OD_PI	OD_P0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-20. SIM Open Drain Configuration Control Register

Table 51-27. SIM Open Drain Configuration Control Register Field Descriptions

Field	Description
31–2	Reserved
1 OD_P1	Open Drain control for Port 1. Used to control whether the XMT data line on port 1 is open-drain. If AMODE bit in SETUP register is set, this bit will have no effect. 0 XMT pin on port 1 is push-pull (default). 1 XMT pin on port 1 is open-drain.
0 OD_P0	Open Drain control for Port 0. Used to control whether the XMT data line on port 0 is open-drain. 0 XMT pin on port 0 is push-pull (default). 1 XMT pin on port 0 is open-drain.

51.3.3.19 SIM Reset Control Register (RESET_CNTL)

See Figure 51-21 for illustration of valid bits in the SIM Reset Control Register and Table 51-28 for description of the bit fields.

0x5001_8050 (RESET_CNTL) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	DBUG	STOP	DOZE	KILL CLOCK	0	FLUSH XMT	FLUSH RCV
W														SOFT RST		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-21. SIM Reset Control Register

Table 51-28. SIM Reset Control Register Field Descriptions

Field	Description
31–7	Reserved
6 DBUG	DEBUG. Used to configure the operation of the SIM module when a debug event occurs. When set, a debug event (generated by such causes as OnCE module and external control) will make the receive FIFO read pointer to be frozen. 0 Debug event has no affect on SIM module (default). 1 Debug event prohibits read pointer changes for receive FIFO.
5 STOP	STOP. Used to configure the operation of the SIM module when a processor STOP instruction is executed. This bit is added to provide support for SIM cards that do not allow the SIM Card clock to be stopped while power is applied. See Figure 51-37. 0 STOP instruction shuts down all SIM clocks (default). 1 STOP instruction shuts down all clocks except for the BAUD_CLK (clock provided to SIM Card).
4 DOZE	DOZE. Used to configure the operation of the SIM module when a processor DOZE instruction is executed. See Figure 51-37. 0 DOZE instruction has no effect on SIM module (default). 1 DOZE instruction will cause SIM module to gate SIM clocks when the transmit FIFO is empty.
3 KILL_CLOCK	Kill SIM Clock. Used to enable/disable the SIM clock input to the SIM module. This bit will gate all SIM clocks including the SIM card clock regardless of the state of the STOP bit described above. 0 SIM input clock enabled (default). 1 SIM input clock disabled.
2 SOFT_RST	Software Reset. Used to reset the entire SIM module. This acts the same as a hardware reset for the SIM module. This bit is self-clearing. **Note:** Software should allow a minimum of 4 reference clock cycles (CKIH) before attempting to access the SIM module after a software reset. 0 SIM Normal operation (default). 1 SIM held in Reset.

Table 51-28. SIM Reset Control Register Field Descriptions (continued)

Field	Description
1 FLUSH_XMT	Flush Transmitter. This bit operates as a SIM transmitter reset. The receive portion of the SIM module is not affected. The software must clear this bit before the SIM transmitter can operate. 0 SIM Transmitter normal operation (default). 1 SIM Transmitter held in Reset.
0 FLUSH_RCV	Flush Receiver. This bit operates as a SIM receiver reset. The transmit portion of the SIM module is not affected. The software must clear this bit before the SIM receiver can operate. 0 SIM Receiver normal operation (default). 1 SIM Receiver held in Reset.

51.3.3.20 SIM Character Wait Time Register (CHAR_WAIT)

See Figure 51-22 for illustration of valid bits in the SIM Character Wait Time Register and Table 51-29 for description of the bit fields.

0x5001_8054 (CHAR_WAIT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CWT[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 51-22. SIM Character Wait Time Register

Table 51-29. SIM Character Wait Time Register Field Descriptions

Field	Description
31–16	Reserved
15–0 CWT	Character Wait Time. The value written to this register will specify the number of ETU times allowed between characters. Default is 0xFFFF

51.3.3.21 SIM General Purpose Counter Register (GPCNT)

See Figure 51-23 for illustration of valid bits in the SIM General Purpose Counter Register and Table 51-30 for description of the bit fields.

0x5001_8058 (GPCNT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPCNT[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 51-23. SIM General Purpose Counter Register

Table 51-30. SIM General Purpose Counter Register Field Descriptions

Field	Description
31–16	Reserved
15–0 GPCNT	General Purpose Counter. The value written to this register will be used to compare to the general purpose counter in the SIM module. Once the General purpose counter reaches this value, the GPCNT flag in the XMT_STATUS register will be set. This counter is intended to be used for any events that must be monitored for duration based on the card clock, receiver sample rate, or ETU rate (transmit clock). Example: ATR arrival time and ATR duration. Default is 0xFFFF

51.3.3.22 SIM Divisor Register (DIVISOR)

See Figure 51-24 for illustration of valid bits in the SIM Divisor Register and Table 51-31 for description of the bit fields.

0x5001_805C (DIVISOR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	DIVISOR[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

Figure 51-24. SIM Divisor Register

Table 51-31. SIM Divisor Register Field Descriptions

Field	Description
31–8	Reserved
7–0 DIVISOR	DIVISOR Register. The value written to this register will be used to generate the SIM Receive clock. The BAUD_SEL[2:0] bits in the CNTL register must be set to ‘111’ in order to control the divisor value using the DIVISOR register. Default is 0xFF

51.3.3.23 SIM Block Wait Time Register (BWT)

See Figure 51-25 for illustration of valid bits in the SIM Block Wait Time Register and Table 51-32 for description of the bit fields.

0x5001_8060 (BWT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWT[15:0]															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 51-25. SIM Block Wait Time Register

Table 51-32. SIM Block Wait Time Register Field Descriptions

Field	Description
31–16	Reserved
15–0 BWT	BWT Register 16 LSB. The value in this register is the block wait time 16 LSB. The time from START bit of last byte sent from the SIM module to the START bit of the first byte sent from the SmartCard must be less than the value formed by the 32-bit register composed by BWT_H and BWT registers. If it is not, then the BWT flag will be set. Default is 0xFFFF

51.3.3.24 SIM Block Guard Time Register (BGT)

See Figure 51-26 for illustration of valid bits in the SIM Block Guard Time Register and Table 51-33 for description of the bit fields.

0x5001_8064 (BGT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BGT[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-26. SIM Block Guard Time Register

Table 51-33. SIM Block Guard Time Register Field Descriptions

Field	Description
31–16	Reserved
15–0 BGT	BGT Register. The value in this register is the block guard time. Time from START bit of last byte sent from the SIM module to the START bit of the first byte sent from the SmartCard must be greater than this value. If it is not, then the BGT flag will be set. Default is 0x0000

51.3.3.25 SIM Block Wait Time Register HIGH (BWT_H)

See Figure 51-27 for illustration of valid bits in the SIM Block Wait Time Register HIGH and Table 51-34 for description of the bit fields.

0x5001_8068 (BWT_H) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BWT_H[15:0]															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 51-27. SIM Block Wait Time Register HIGH

Table 51-34. SIM Block Wait Time Register HIGH Field Descriptions

Field	Description
31–16	Reserved
15–0 BWT_H	BWT Register 16 MSB. The value in this register is the block wait time 16 MSB. The time from START bit of last byte sent from the SIM module to the START bit of the first byte sent from the SmartCard must be less than the value formed by the 32-bit register composed by BWT_H and BWT registers. If it is not, then the BWT flag is set. Default is 0xFFFF

51.3.3.26 SIM Transmit FIFO Status Register (XMT_FIFO_STAT)

See Figure 51-28 for illustration of valid bits in the SIM Transmit FIFO Status Register and Table 51-35 for description of the bit fields.

0x5001_806C (XMT_FIFO_STAT) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	XMT_CNT[3:0]				XMT_WPTR[3:0]				XMT_RPTR[3:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-28. SIM Transmit FIFO Status Register

Table 51-35. SIM Transmit FIFO Status Register Field Descriptions

Field	Description
31–12	Reserved
11–8 XMT_CNT	These bits indicate the number of Bytes in the transmit FIFO. ‘0’ value means FIFO is empty or full.
7–4 XMT_WPTR	These bits indicate the transmit FIFO Write Pointer.
3–0 XMT_RPTR	These bits indicate the transmit FIFO Read Pointer.

51.3.3.27 SIM Receive FIFO Counter Register (RCV_FIFO_CNT)

See Figure 51-29 for illustration of valid bits in the SIM Receiver FIFO Counter Register and Table 51-36 for description of the bit fields.

0x5001_8070 (RCV_FIFO_CNT) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	RCV_CNT[8:0]								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-29. SIM Receive FIFO Counter Register

Table 51-36. SIM Receive FIFO Counter Register Field Descriptions

Field	Description
31–9	Reserved
8–0 RCV_CNT	These bits indicate the number of Byte can be written into the receive FIFO. 0 value means FIFO is empty or full.

51.3.3.28 SIM Receive FIFO Write Pointer Register (RCV_FIFO_WPTR)

See Figure 51-30 for illustration of valid bits in the SIM Receive FIFO Write Pointer Register and Table 51-37 for description of the bit fields.

0x5001_8074 (RCV_FIFO_WPTR) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	RCV_WPTR[8:0]								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-30. SIM Receive FIFO Write Pointer Register

Table 51-37. SIM Receive FIFO Write Pointer Register Field Descriptions

Field	Description
31–9	Reserved
8–0 RCV_WPTR	These bits indicate the receive FIFO Write pointer.

51.3.3.29 SIM Receive FIFO Read Pointer Register (RCV_FIFO_RPTR)

See Figure 51-31 for illustration of valid bits in the SIM Receive FIFO Read Pointer Register and Table 51-38 for description of the bit fields.

0x5001_8078 (RCV_FIFO_RPTR) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	RCV_RPTR[8:0]								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 51-31. SIM Receive FIFO Read Pointer Register

Table 51-38. SIM Receive FIFO Read Pointer Register Field Descriptions

Field	Description
31–9	Reserved
8–0 RCV_RPTR	These bits indicate the receiver FIFO read pointer.

51.4 Functional Description

To best describe the organization of the SIM module from a user's point of view, it is instructive to view the SIM at a number of different levels of hierarchy. See Figure 51-32 for illustration of the organization of SIM and connection of the signals to the two available serial ports.

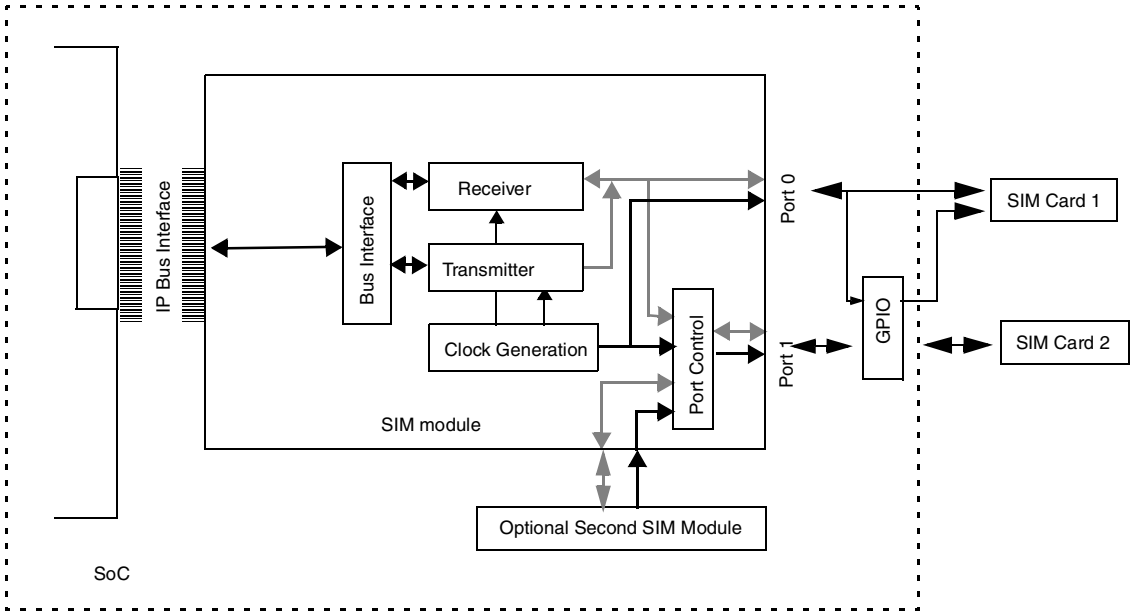


Figure 51-32. Block Diagram for SIM Module

The SIM module is essentially a standard UART with some special provisions made for SIM card communication. The SIM consists of nine main parts:

- IP Bus Interface
- Bus interface
- Clock generator
- Transmitter
- Receiver
- Port controller
- General purpose counter
- LRC blocks
- CRC blocks

See Figure 51-33 for illustration of the block diagram in detail, specifically the nine main parts.

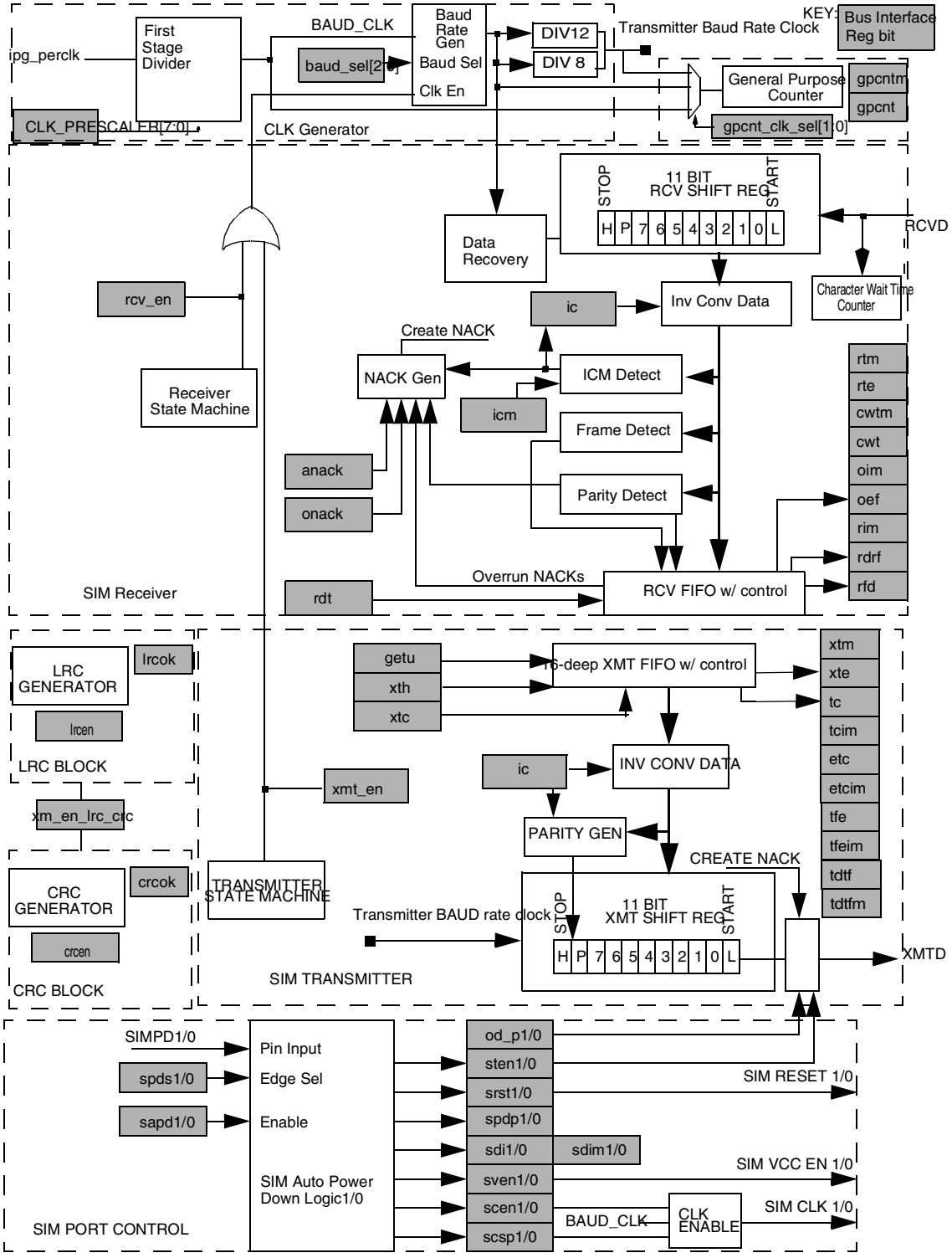


Figure 51-33. SIM Detailed Block Diagram

51.4.1 SIM Bus Interface

The SIM Bus interface block has been designed to enable the SIM module to be easily ported to other MCU cores. The bus interface block contains all of the logic that uses the bus interface signals. See Figure 51-34.

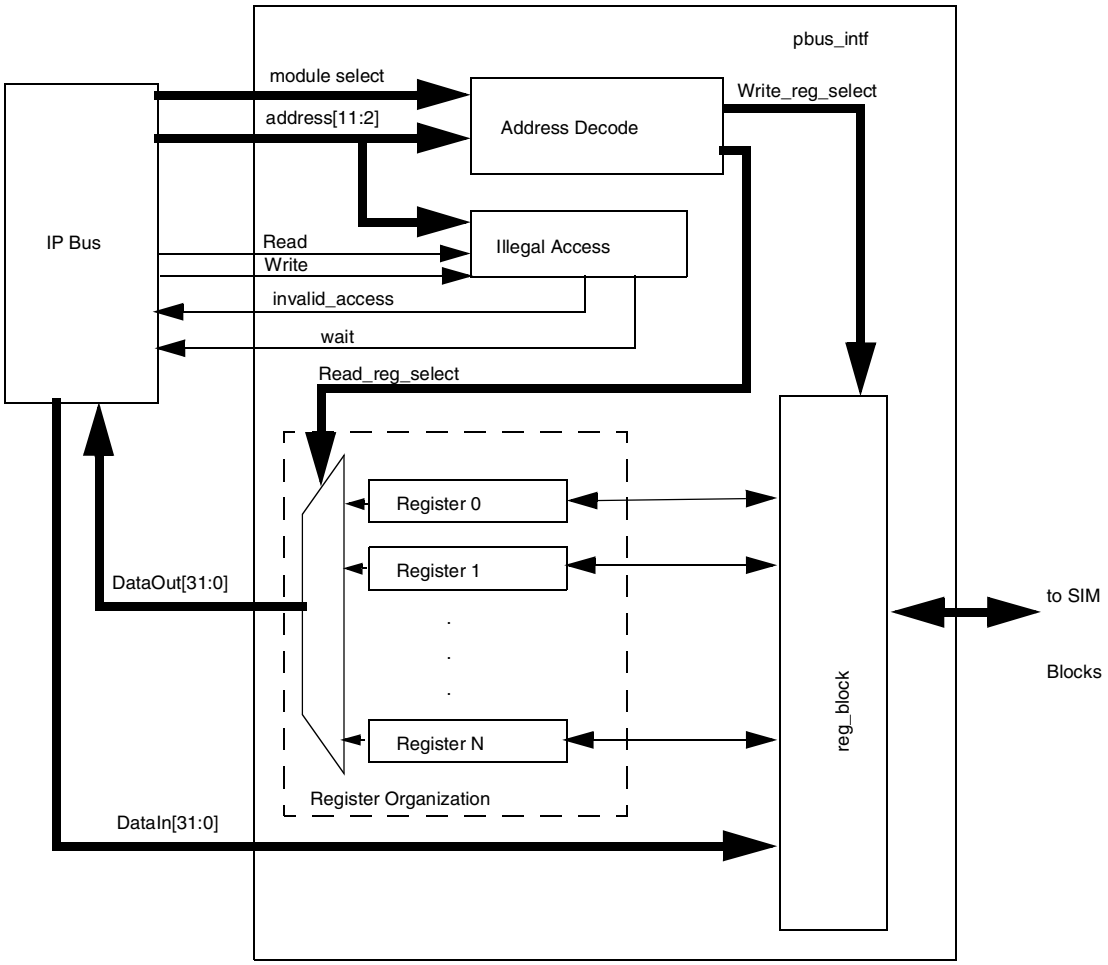


Figure 51-34. SIM Bus Interface

The bus interface block is responsible for peripheral bus address decoding, illegal access detection, dynamic wait-state requests, register organization, and register bit implementations. The address decoding uses the module address bus input along with the module select output of the IP bus to decode if the SIM module is being addressed. This decoding of these signals is done asynchronously. The output of the address decode is used with the read/write signals from the IP bus to determine the action requested by the bus master. If the read signal is active, the data_out bus will be driven with the register selected by the address decoding. If the write signal is active, the data_in bus will be latched into the register selected by the address decoder at the rising edge of the data_strobe signal.

The illegal access detection is implemented similarly to the address decoding. If the module_select signal becomes active while the address inputs are pointing to an unimplemented address, the invalid signal is

asserted asynchronously to the bus master. The invalid signal can also be generated under certain register access conditions such as writing to a full transmit FIFO, or writing to a read-only register. See Figure 51-35.

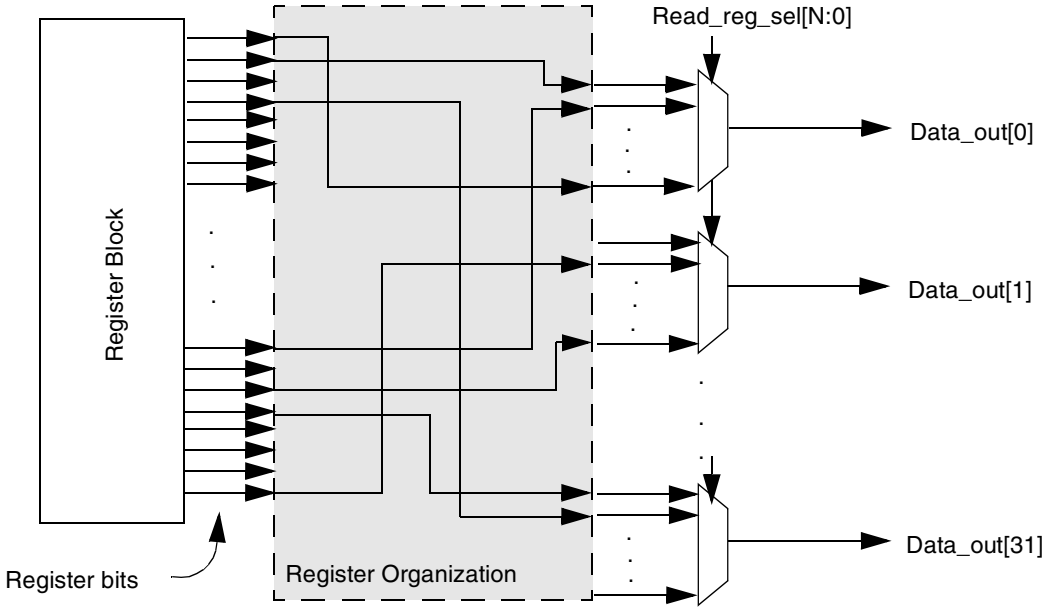


Figure 51-35. SIM Read Registers

The register bit implementations are done inside the reg_block sub-module. This block accepts the decoded read and write signals for the register bits generated in the address decoder. The data_strobe output is used as the clock input to the register bits in order to clock in the new data during a write access. The write_reg_sel signals are used to gate the write action with the address decoding. The clearing mechanisms for status bits are implemented as write-one-to-clear. See Figure 51-36.

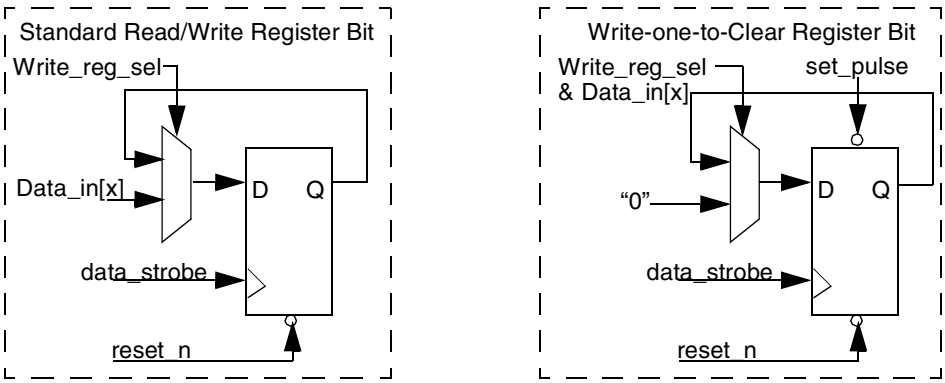


Figure 51-36. Register Bit Diagram

In the "write-one-to-clear" register bit example, "set_pulse" represents a pulsed signal generated in the data_strobe clock domain to set the register output.

51.4.2 SIM Clock Generator

The clock generator is responsible for implementing clock tree synthesis constructs, scan clock muxing, baud rate clock (BAUD_CLK) generation, and providing clocks to the transmitter, receiver, and port controller sections of the SIM module. See Figure 51-37 for the schematic of the SIM clock generator. The dividers outlined in bold, generate a pulsed (gated) clock of the desired frequency. The pulse is equal in duration to one half the ipg_perclk period. This clock is used internal to the SIM module to drive the transmit and receive sections of the module.

The dividers outline in normal, generate a clock of 45%~55% duty cycle to drive the clock pin of the SmartCard. This clock is not used internal to the SIM module.

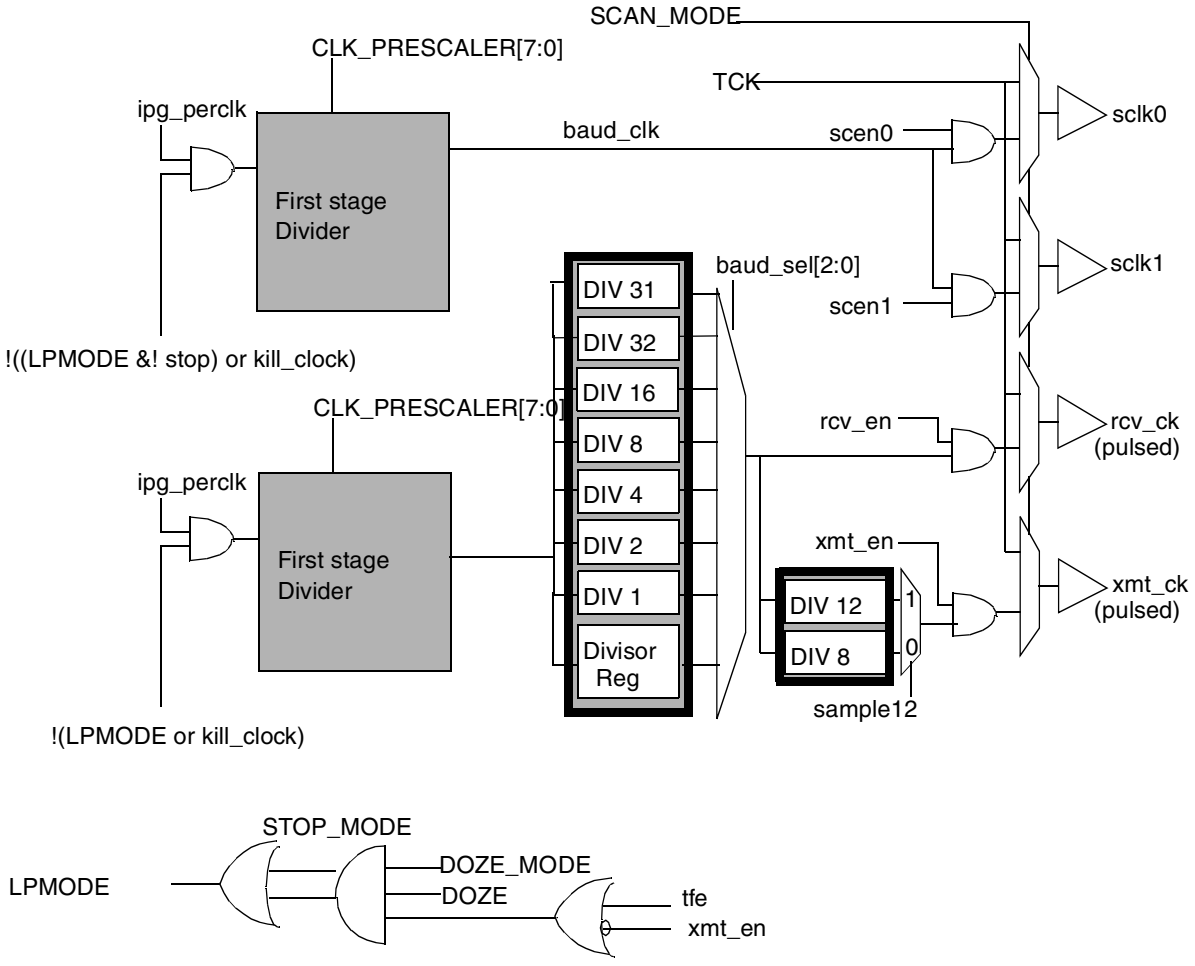


Figure 51-37. SIM Clock Generator

51.4.2.1 Clock Tree Synthesis

The clock tree synthesis constructs used in the current implementation follow the current clock methodology(srs3.1 clock methodology). This can be seen in the RTL of the sim_clocks.v module. There

is an instance of the special cell (clk_driver_pos.v) to drive each clock. This cell can then be used to control the clock skew during the layout process.

51.4.2.2 Scan Test

The scan clock muxing is implemented according to the design for testability rules.

51.4.2.3 Baud Clock Generation

The baud rate clock generation performed by the clock generator results in different frequencies. The default frequency is a divide by two of the ipg_perclk input. The baud rate can be programmed to be ipg_perclk divided by the even value using the CLK_PRESCALER[7:0] bits as shown in the CLK_PRESCALER register. The baud rate clock is generated in two forms in the design. The BAUD_CLK that is used by the SIM cards (SCLK0, SCLK1) is generated so that it must be 45%~55% duty at the divide values. This is necessary to meet the requirements of the ISO 7816 specification. The BAUD_CLK that is used internal to the SIM module is generated as a gated version of the ipg_perclk clock. The resultant clock will be a pulse of one-half ipg_perclk period in width with the expected frequency. The gated baud clock complies with the clock methodology initiative. The 50% duty cycle clock does not comply with the clock methodology initiative but this clock is not used as a clock internal to the SIM module. It is only used as a data path.

51.4.2.4 Transmitter Clock Generation

The transmitter clock (xmt_clk) is generated by the clock generator based on the values passed to it for the baud rate select (baud_sel[2:0]) and sample12. The transmit clock is gated by the transmit enable (XMT_EN) register bit. When the transmitter is enabled, the clock generator counts the appropriate number of receive clock (rcv_clk) positive edges to determine when to toggle the transmitter clock output. The transmitter clock is always based upon the receive clock.

51.4.2.5 Receiver Clock Generation

The receiver clock (rcv_clk) is generated by the clock generator based on the value passed to it for the baud rate select (baud_sel[2:0]). The receiver clock is gated by the receiver enable (RCV_EN) register bit. When the receiver is enabled, the clock generator counts the appropriate number of BAUD_CLK positive edges to determine when to toggle the receiver clock output. The number of BAUD_CLK positive edges is determined by the value of the baud rate select input (baud_sel[2:0]) and is programmable to these divisors: 31 (slowest because used with the 372/1 Fi/Di rate), 32, 16, 8, 4, 2, 1, and divisor[7:0]. The programmable divisor (divisor[7:0]) is programmable via the DIVISOR_REG register.

51.4.2.6 Port Control Clock Generation

The port controller clocks are provided by the clock generator for use on the SIM card ports. These clocks are equivalent in frequency to the BAUD_CLK and are gated by the SIM clock enable (scen0 and scen1) signals. The level at which the card clocks (SCLK0, SCLK1) are stopped when disabled is determined by the SIM clock select polarity (SCSP0, SCSP1) inputs to the clock generator. Synchronizers are

implemented to ensure glitch free operation of the card clocks when enabling/disabling, or changing the clock stopped polarity.

51.4.2.7 Low Power Mode Clock Control

The clock generator block is responsible for gating the clocks to the SIM module appropriately whenever a low power mode instruction is decoded. The IP interface provides two signals that encode the two available low-power states of the processor. These states are: STOP and DOZE. The response to the DOZE instruction is controlled through the use of the DOZE bit in the RESET_CNTL register. See the RESET_CNTL register description. The response to the STOP instruction is controlled through the use of the STOP bit in the RESET_CNTL register. See the RESET_CNTL description

51.4.3 SIM Transmitter

The SIM Transmitter block has been designed to localize all transmit related circuitry into one section of hierarchy. The Transmitter block comprises the following sections of logic: transmit state machine, transmit shift register, transmit FIFO, guard time generator, transmit NACK control, and transmit data convention.

51.4.3.1 Transmit State Machine

The Transmit state machine is the heart of the transmitter block. The state machine is responsible for sequencing through a transmit operation while reacting to inputs from the receiver, the transmit FIFO, and the guard time circuit. See Figure 51-38 for flow diagram of the transmit state machine.

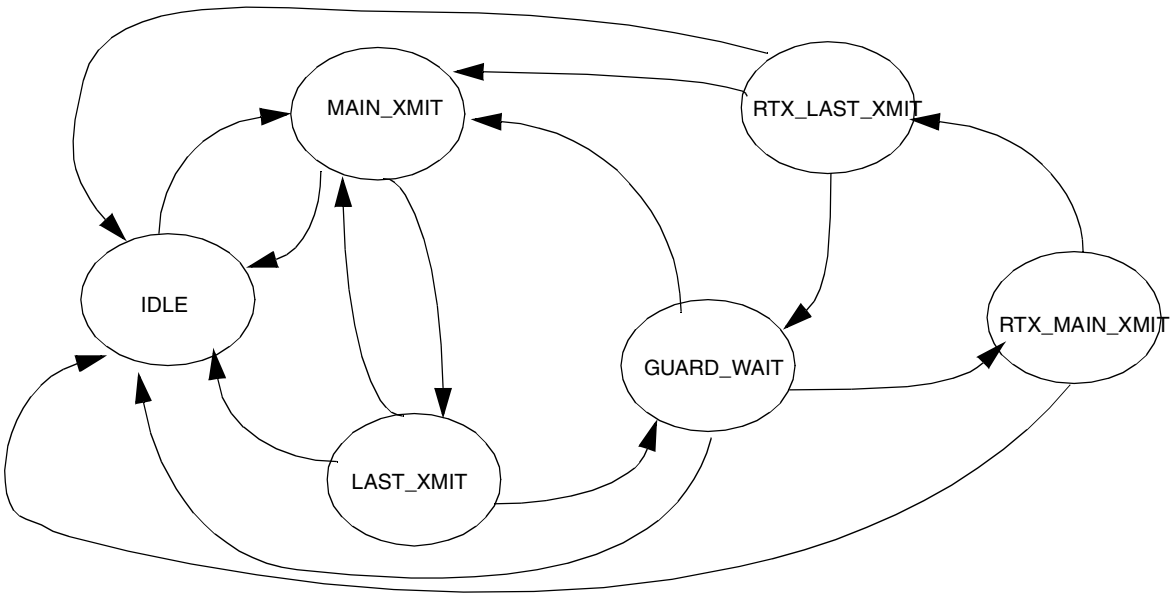


Figure 51-38. Transmit State Machine

The functions performed by each state are as follows:

- IDLE
  - This is the initial state. The state machine waits here until it detects XMT_EN is set and a write to the transmit FIFO has occurred. The data pointed to by the transmit read pointer is loaded into the shift register, and the state machine transitions to the MAIN_XMIT state. Any time XMT_EN is cleared, the state machine returns to this state.
- MAIN_XMIT
  - The transmitter is operating normally in this state. The data in the shift register is shifting once every transmit clock cycle. When the second to last bit of the current transmission is about to be sent, the state machine transitions to the LAST_XMIT state.
- LAST_XMIT
  - This state transmits the last bit of the current transmission and determines the next operation. One of the following will occur.
    - If GETU[7:0] is non-zero, jump to the GUARD_WAIT state.
    - If a transmit NACK error occurred, with a zero in GETU[7:0], jump to MAIN_XMIT state to retransmit the current byte.
    - If no transmit NACK, and GETU[7:0] is zero, load the shift register, jump to MAIN_XMIT to transmit the next byte
    - If no transmit NACK, GETU[7:0] is zero, and the FIFO is empty, jump to IDLE state; set the transmit complete (TC) flag.
- GUARD_WAIT
  - The state machine remains in this state until the Guard time counter has expired.
    - If a transmit NACK error occurred on last transmission, jump to RTX_MAIN_XMIT and re-transmit
    - If no transmit NACK and the FIFO is not empty, load the shift register, jump to MAIN_XMIT to transmit the next byte.
    - If no transmit NACK and the FIFO is empty, return to the IDLE state.
    - If transmit NACK threshold is detected, stop transmitter, set XTE flag, and jump to the IDLE state.
- RTX_MAIN_XMIT
  - The transmitter is operating normally in this state. The data in the shift register is shifting once every transmit clock cycle. When the second to last bit of the current transmission is about to be sent, the state machine transitions to the RTX_LAST_XMIT state. This state is identical to the MAIN_XMIT state except that it retransmits the previously NACKed byte.
- RTX_LAST_XMIT
  - This state transmits the last bit of the current re-transmission and determines the next operation. One of the following will occur.
    - If GETU[7:0] is non-zero, jump to the GUARD_WAIT state.
    - If a transmit NACK error occurred, with a zero in GETU[7:0], jump to GUARD_WAIT state to check Transmit NACK threshold.

- If no transmit NACK, GETU[7:0] is zero, and the FIFO is not empty, load the shift register, jump to MAIN_XMIT to transmit the next byte
- If no transmit NACK, GETU[7:0] is zero, and the FIFO is empty, jump to IDLE state; set the transmit complete (TC) flag.

51.4.3.2 Transmit Shift Register

The transmit shift register is 11 bits wide and controlled by the transmit state machine described previously. The shift register shifts out data at the transmit clock frequency (xmt_ck).

51.4.3.3 Transmit FIFO

The transmit FIFO is implemented inside the transmitter block. The FIFO depth is 16 bytes. The FIFO block is shared by both SIM module ports. The transmit FIFO cannot be accessed by the alternate SIM module through the alternate port. Each write to the transmit FIFO increases the transmit FIFO write pointer. Each time the transmit shift register is loaded from the transmit FIFO, the transmit FIFO read pointer is increased. When the read and write pointers are equal, the transmit FIFO empty flag (TFE) is set. Software has no visibility of the transmit FIFO pointers, but a transmit FIFO threshold value can be set to alert the software when the number of bytes in the FIFO has reached a specified level. A read of the transmit FIFO register (XMT_BUF) will return the last byte written to the FIFO. Writes to the transmit FIFO can occur at any time.

The transmit FIFO can be flushed by setting the XMT_FLUSH bit in the RESET_CNTL register. A transmit NACK threshold error (XTE) will halt the transmitter, and flush the transmit FIFO. The flush operation resets the transmit read and write pointers to equal values. Everything in the transmitter block is reset by the transmit flush operation. This does not include the control registers associated with the transmitter. The Transmit data threshold flag (TDTF) does not get cleared by the XMT_FLUSH operation.

51.4.3.4 Transmit Guard Time Generator

The guard time generator is simply a counter that is clocked by the transmit clock in order to delay the beginning of the next transmission and the setting of the transmit complete interrupt flag (TC) by a programmable amount of transmit bit widths (ETU’s). The duration of the count is controlled by the

GUARD_CNTL register (GETU[7:0]). See Figure 51-39 for depiction of three transmit operations in order to show the effect of the guard time generator logic

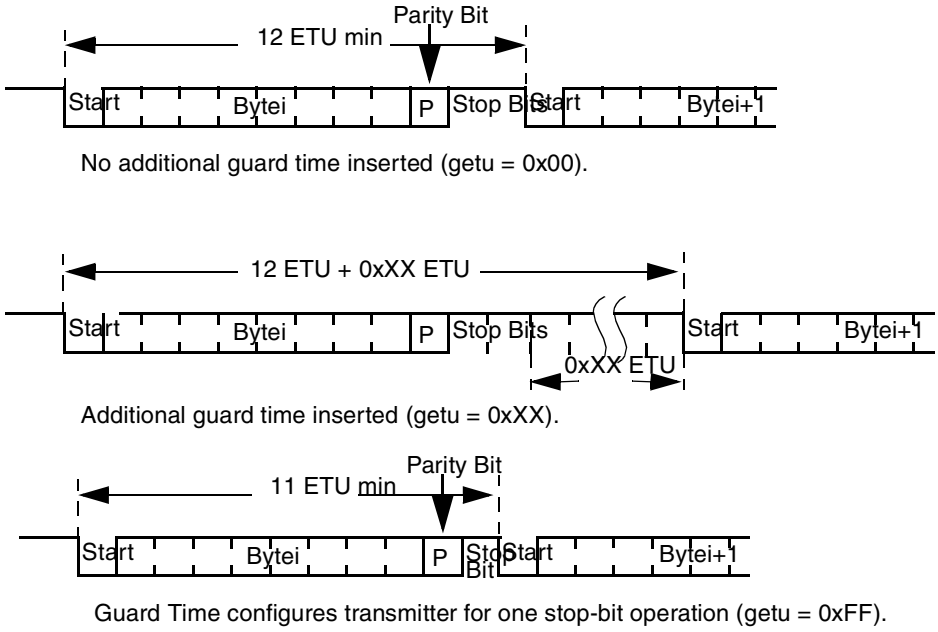


Figure 51-39. Transmit Guard Time

51.4.3.5 Transmit NACK Generator

The transmit NACK generator is responsible for driving the transmitter output low during the STOP bit time to signify an error was detected in the received data from the SIM card. This logic responds to a NACK request generated by the receiver block. See Figure 51-40 shows a typical SIM transaction with the NACK pulse inserted.

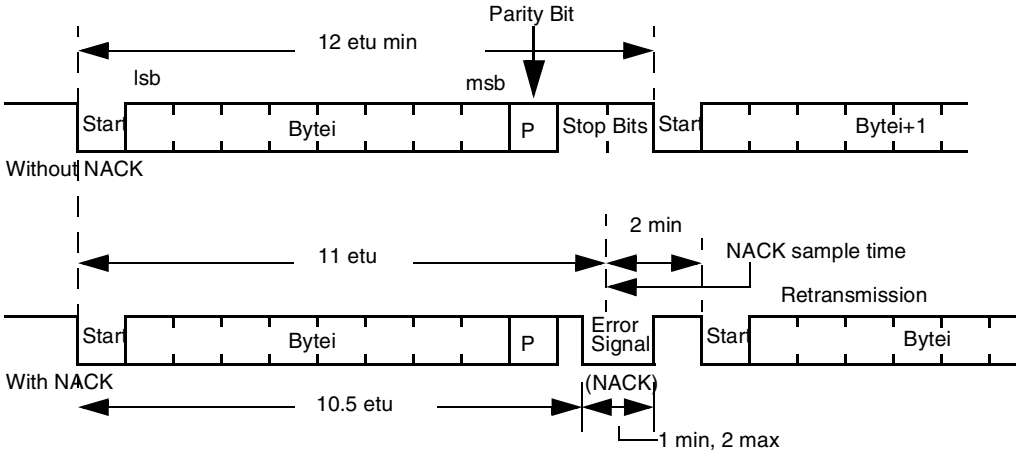


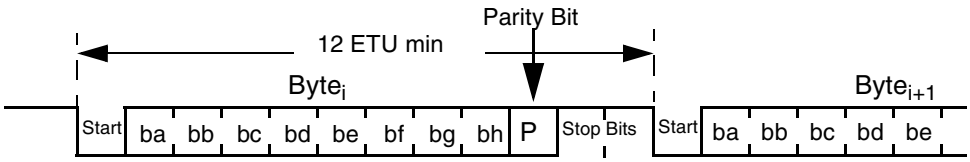
Figure 51-40. Transmit NACK Generator

The transmit NACK generator is also responsible for keeping track of the number of NACKs received during a transmit operation. The SIM receive state machine detects NACKs generated by the SIM card,

and reports them to the transmit NACK logic. Once the number of detected NACKs has reached the programmed threshold (XTH[3:0]), an interrupt flag is generated, the transmit FIFO is flushed, and the transmitter is disabled.

51.4.3.6 Transmit Data Convention Logic

The transmit data convention logic provides the support for the two different data conventions available in SIM cards. See Figure 51-41 for illustration of SIM data conventions.



Parity Bit: if configured for even parity, total number of logic ones in the 9 bits (8 bits data, 1 parity bit) will be even
if configured for odd parity, total number of logic ones in the 9 bits (8 bits data, 1 parity bit) will be odd
When configured for inverse convention, the parity bit is inverted by SIM before being transmitted.

Direct Convention: *ba* is lsb of data byte to be sent. *bh* is msb.
Neither the data bits nor parity bit is logically inverted.

Inverse Convention: *ba* is msb of data byte to be sent. *bh* is lsb.
Both the data bits and parity bit are logically inverted by hardware.

Figure 51-41. SIM Data Conventions

The direct data convention is the default. If the inverse convention bit (IC) in the DATA_FORMAT register is set, the transmit data convention logic converts the output of the transmit FIFO to the inverse convention before sending it to the transmit shift register.

51.4.4 SIM Receiver

The SIM Receiver block has been designed to localize all receive related circuitry into one section of hierarchy. The receiver block is comprised of the following functions: receive state machine and the receive FIFO.

51.4.4.1 Receive State Machine

The receive state machine is responsible for sampling the receive data pin and capturing the bit value into the receive shift register. Additionally, the receive state machine can detect the START bit, parity errors, framing errors, and initial character when operating in initial character mode.

Once enabled by the RCV_EN bit in the ENABLE register, the receive state machine sequences through the states as shown in Figure 51-42.

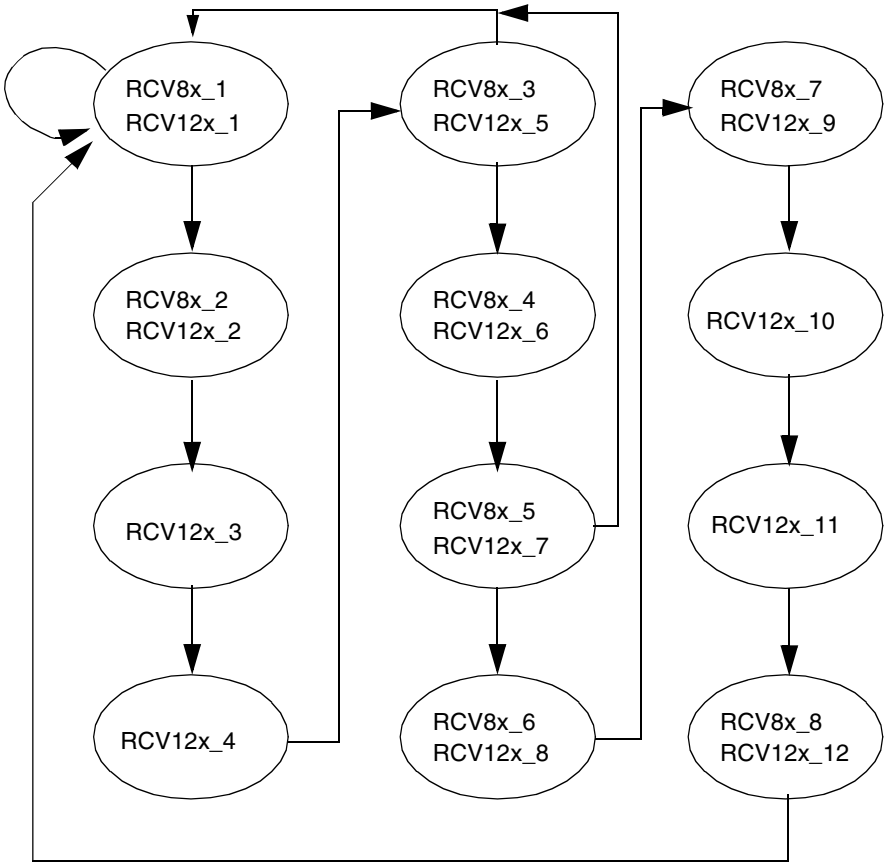


Figure 51-42. Receive State Machine

The states identified with a “8x” are used when operating in a 8x oversampling mode. The states identified with a “12x” are used when operating in a 12x oversampling mode. This mode is only used when sample12 is set to “1”. The number following the oversampling mode identifier represents the state number in the current mode. There are 12 states in “12x” mode, and 8 states in “8x” mode. Some states simply implement a one RCV_CK delay. States that perform additional functions are:

- RCV8x_1, RCV12x_1
  - This is the initial state of the receive state machine. If the first bit has not been received, the state machine remains in this state until a valid START bit is detected. For every subsequent bit, this state is simply a one RCV_CK cycle delay.
- RCV8x_2, RCV12x_2
  - This state captures the first sample of the current receive data input.
- RCV12x_3
  - This state captures the second sample of the current receive data input.

- RCV12x_4
  - This state captures the third sample of the current receive data input.
- RCV8x_3, RCV12x_5
  - This state checks if we are receiving a correct START Bit. If not, Then we return to state 1.
- RCV8x_4, RCV12x_6
  - If we are in the 11th Bit, We check for Parity or Initial Character errors and send NACK if needed.
- RCV8x_5, RCV12x_7
  - Store current Bit value in Shift Register. If this is the first bit and the value is not 0, restart the State Machine.
- RCV8x_6, RCV12x_8
  - If the current bit is the last bit of the transfer, Set flag to transfer Shift Register to Receive Buffer.
- RCV8x_7, RCV12x_9
  - Clear flag for transferring Shift Register to Receive Buffer.
- RCV12x_10
  - If the current bit is the last bit of the transfer (first stop bit), this state takes the sample of the NACK window.
- RCV12x_11
  - If the current bit is the last bit of the transfer (first stop bit), this state takes the sample of the NACK window.
- RCV8x_8, RCV12x_12
  - This state represents the end of the current receive input bit. Several operations occur during this state, including:
    - Increment bit counter
    - Perform a majority vote on the NACK samples and notify the transmitter if a NACK pulse was detected.

51.4.4.2 Data Sampling/Voting

The receive state machine runs at the receive clock rate (RCV_CK). This clock is used to oversample the received data at either a 8X or 12X sample rate. For each input bit, the receive state machine captures three samples. A majority voting algorithm is then applied to determine the value of the bit received. The value common to two or more of the samples is determined to be the bit value in the receive shift register.

51.4.4.3 Start Bit Detection

The SIM receive input is defined to be high when not active. The data transmission is defined to begin with a low pulse for a bit duration. This is called the start bit. The receive state machine is responsible for detecting and validating the start bit. The receive state machine samples the start bit three times using a

majority voting scheme to determine if the start bit is valid. This effectively filters out any low receive inputs shorter than one RCV_CK period.

See Figure 51-43 for illustration of a typical SIM data transaction with the START bit identified.

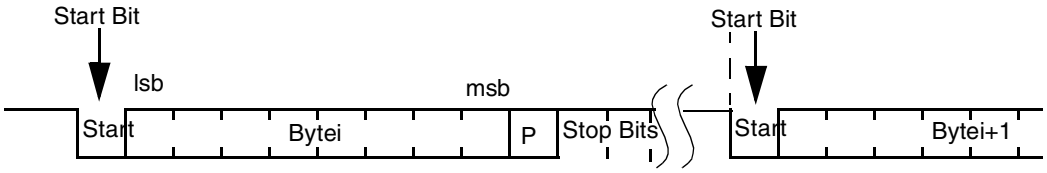


Figure 51-43. Start Bit Diagram

51.4.4.4 Parity Error Detection

The receive state machine is responsible for detecting parity errors in the received data. Data is always transmitted with even parity, except when in inverse convention mode. In inverse convention mode, all data bits and the parity bit are complemented making the data appear to be odd parity. The parity bit is defined as the tenth bit of the received data. The parity of the second through the tenth received bit is calculated by the receiver parity logic. This logic determines if the parity of the nine received bits is correct.

See Figure 51-44 for illustration of a typical SIM data transaction with the parity bit identified.

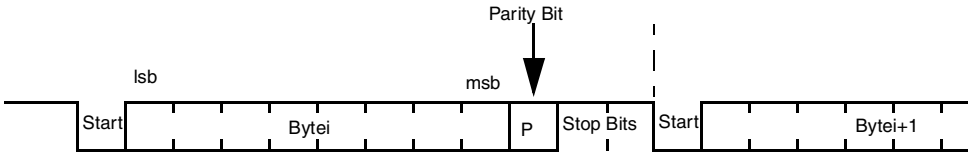


Figure 51-44. Parity Bit Diagram

When a parity error is detected on a given byte, the RCV_PF bit for that byte is set in the receive FIFO. A parity error cannot cause an interrupt, but it can signal the SIM transmitter to create a NACK pulse to the SIM card asking for a retransmission of the corrupted data. NACK generation upon a parity error is enabled by setting the ANACK bit in the CNTL register.

51.4.4.5 Framing Error Detection

The receive state machine is responsible for detecting framing errors in the received data. A SIM data transaction is defined as 11 or 12 bits long consisting of the start bit, 8 data bits, 1 parity bit and 1 or 2 stop bits. A framing error occurs when the stop bit is not detected during the 11th bit time of a data transaction. The stop bit is generally defined as two bit times (ETU's) of a high pulse following the parity bit. When the GUARD_CNTL register is programmed to 0xFF, the stop bit is defined as one bit time. A framing error can only occur when the parity bit of the current byte is low, and the stop bit arrives late.

See Figure 51-45 for illustration of a typical SIM data transaction with the stop bits identified. Also, shown is a SIM data transaction with a late arriving stop bit indicating a framing error.

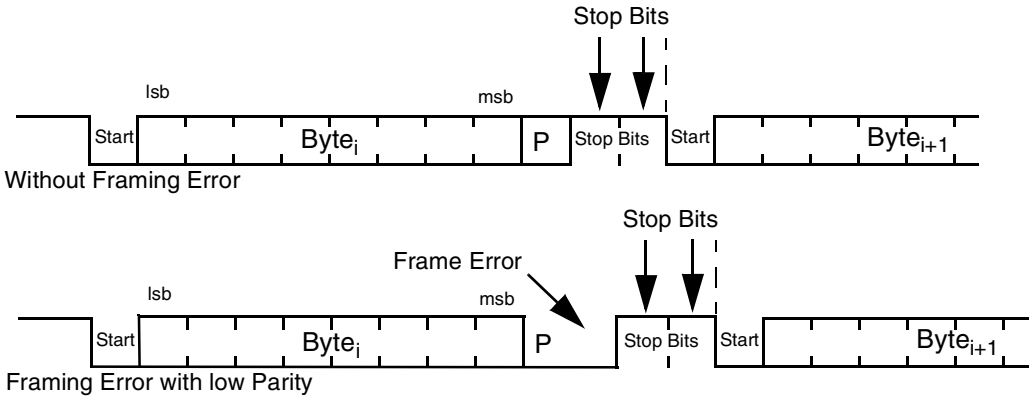


Figure 51-45. Framing Error Diagram

When a framing error is detected on a given byte, the RCV_FE bit for that byte is set in the receive FIFO. A framing error cannot cause an interrupt, nor can it create a NACK pulse to the SIM card asking for a retransmission of the corrupted data.

51.4.4.6 NACK Detection

The existence of the NACK pulse is sampled by the receive state machine at 11 Elementary Time Units (ETUs) after the falling edge of the START bit. An ETU is equivalent in time to 1 transmit clock period. Once the receiver detects a NACK, it signals the transmitter that an error occurred. The transmitter will not initiate retransmission for at least another two ETU times as required by the ISO 7816 specification.

51.4.4.7 Initial Character Detection

The SIM receive state machine supports the detection of special characters that allow it to determine what data format is being used by the connected SIM card. When placed in initial character mode, the SIM expects to receive one of two potential characters that it will use to set the data format control bit, IC, in the DATA_FORMAT register.

The two possible data formats are inverse convention and direct convention. Figure 51-46 and Figure 51-47 illustrate the differences between the two formats. Essentially, inverse convention differs from direct convention in that the order of the data is flipped msb for lsb and the data bits and parity bit

are logically inverted. When receiving inverse convention data, the transformation of the data back to direct convention format is done in hardware, including the inversion of the data and parity bits.

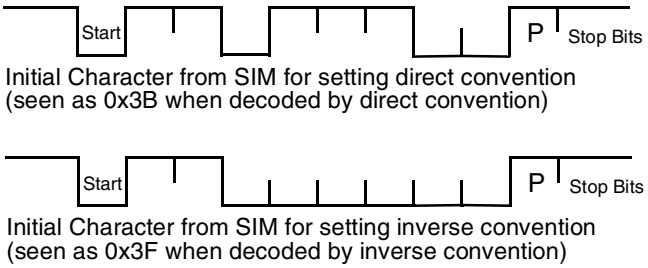
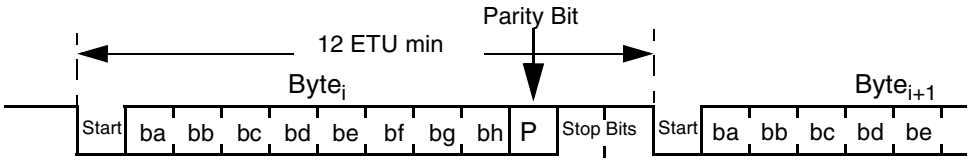


Figure 51-46. Valid Initial Characters



Parity Bit: if configured for even parity, total number of logic ones in the 9 bits (8 bits data, 1 parity bit) will be even
if configured for odd parity, total number of logic ones in the 9 bits (8 bits data, 1 parity bit) will be odd
When configured for inverse convention, the parity bit is inverted by SIM card before being transmitted.

Direct Convention: *ba* is lsb of data byte to be sent. *bh* is msb.
Neither the data bits nor parity bit is logically inverted.

Inverse Convention: *ba* is msb of data byte to be sent. *bh* is lsb.
Both the data bits and parity bit are logically inverted by SIM card.

Figure 51-47. Inverse Convention versus Direct Convention

51.4.4.8 Receive FIFO

The receive FIFO is implemented inside a sub-block of the receiver block. The FIFO depth is 288 bytes. The FIFO block is shared by both SIM module ports. The receive FIFO cannot be accessed by another SIM module through the alternate port. Only the port1 IO pins can be accessed through the alternate port. The secondary SIM would then uses its own internal FIFO while using the primary SIM IO ports. This is only done if it is required to control two SIM cards at the same time. It the two SIM card access only occurs one at a time, then a single SIM module can control both SIM cards (one on port0 and one on port1). The receive FIFO is accessed through the RCV_BUF register.

The receive FIFO is loaded from the receive shift register after the final bit of the current SIM card transmission has been received. The FIFO contains 10 bits per transmission. The lower eight bits contain the received data byte. Bits 8 and 9 contain the parity and framing status for the received byte.

Each read from the receive FIFO increases the receive FIFO read pointer. Each time the receive shift register is transferred to the receive FIFO, the receive FIFO write pointer is increased. When the difference between the read and write pointers equals the programmed threshold value (RDT), the receive data register full flag (RDRF) will be set. An interrupt can be generated by the RDRF flag if the RIM bit in the INT_MASK register is cleared. Software has no visibility of the receive FIFO pointers. A write to the receive FIFO register (RCV_BUF) will generate an invalid access exception to the processor.

The receive FIFO can be flushed by setting the RCV_FLUSH bit in the RESET_CNTL register. The flush operation resets the receive read and write pointers to equal values. All logic associated with the receiver will be reset by the flush operation except for receiver control registers.

51.4.4.9 Overrun Detection

The receive FIFO logic is responsible for detecting an overrun condition. When a received byte is transferred from the receive shift register to a receive FIFO that contains 288 unread bytes, the SIM receiver will flag an overrun condition. This condition will always set the overrun error flag, OEF in the RCV_STATUS register. The received byte will be discarded leaving the 288 unread bytes in the FIFO unaltered. The SIM module will generate a NACK to the SIM card on an overrun condition if the ONACK bit in the CNTL register is set. The SIM module will continually NACK SIM card transmissions until a read of the receive FIFO occurs.

51.4.4.10 Character Wait Time Counter

The SIM receiver block includes a 16-bit counter that counts the number of bit times (ETUs) between received characters. When enabled, the Character Wait Time Counter (CWT) will not start counting until after the START bit(s) of a valid character has been received. The counter is synchronized to the receive character bit positions so that an accurate count of the number of ETUs between characters can be made. The Character Wait Time Counter has a 16-bit programmable comparator. Software can write the expected number of ETUs between characters to the comparator. If the time between characters exceeds this value, an interrupt flag will be set and an interrupt generated if the mask is clear.

51.4.5 SIM Port Control

The SIM Port Control block has been designed to localize all port related circuitry into one section of hierarchy. The port control block comprises the following functions: SIM card interface, SIM Card presence detect, and SIM card auto-power down.

51.4.5.1 SIM Card Interface

The SIM module allows for direct control of two separate SIM cards. The SIM module does not support simultaneous communication with two SIM cards. To achieve simultaneous communication with two SIM cards, a second SIM module must be used. The SIM module can relinquish control of the inactive port to a secondary SIM module by setting the AMODE bit in the SETUP register. When alternate SIM Card Mode enable, output signals in port1 is directly from the secondary SIM module and input signals in port1 will pass through to the secondary SIM module.

The SIM card clock is generated in the SIM clock generator. The SIM card reset and SIM card voltage enable are controlled by software through the PORTx_CNTL registers.

See Figure 51-48 for illustration of a example SIM module hookup to two SIM cards. The power management chip shown is used to provide Vcc for the SIM card, and level translators for the remaining signals when interfacing to a SIM card operating at a different voltage than SIM module. The SIM module can directly access a SIM card if it is operating at the same voltage. In this case, the 3VOLTx bit in the PORTx_CNTL registers can be configured so the SIM Port transmit output as bi-directional. This frees up

the SIM port receive pin to be used as general purpose I/O. If the SIM module and the SIM card are operating at different voltages, then the power management level translators must be used. In this case, the 3VOLTx bit must be cleared and then both the RX and TX pins will be used.

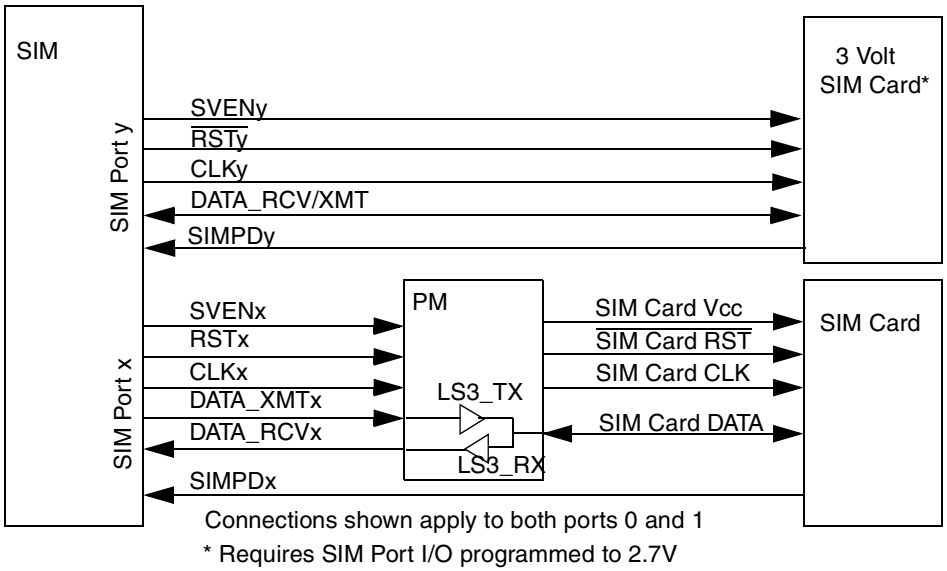


Figure 51-48. SIM Card Hookup

51.4.5.2 SIM Card Presence Detect

The SIMPDx input allows for detection of the insertion or removal of a SIM card. The SPDSx control bit in the PORTx_DETECT register allows the software to configure which edge of the SIMPDx pin will cause a presence detect event. A maskable interrupt can be generated when a SIMPDx event occurs.

An internal pull-down device is present on the SIMPDx pins. This will provide for a high to low transition on the SIMPDx pin when a SIM card is removed.

51.4.5.3 SIM Card Automatic Power Down

When interfacing to the SIM cards, it is necessary to follow a particular sequence when powering them up and down. The SIM port control block contains hardware that provides the correct sequence to power down a SIM card (see Figure 51-49). The power up sequence must be done manually by the software using the pin control bits supplied in the PORTx_CNTL registers.

The power down sequence is specified in ISO 7816 as follows:

1. RST transitions from high to low.
2. CLK is turned off to a low.
3. I/O transitions from high impedance to low.
4. SIM Vcc is turned off.

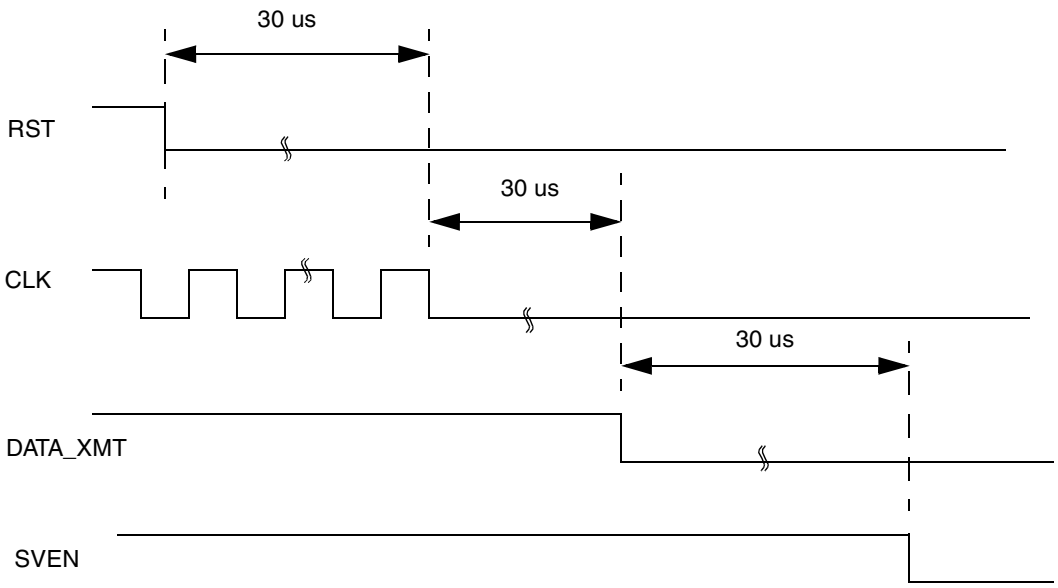


Figure 51-49. Auto Power Down Sequence

51.4.6 SIM General Purpose Counter

The SIM module provides a 16-bit counter for use when timing events during SIM card communication. The clock source for the counter is selectable between three sources: BAUD_CLK, RCV_CLK, or XMT_CK (ETU clock). The GPCNT_CLKSEL[1:0] bits in the CNTL register are used to select the clock input. The counter is enabled as soon as the input clock is selected. The starting of the counter is immediate once the input clock is running. Software can control the three input clock sources by using the KILL_CLOCK, RCV_EN and XMT_EN bits provided in the RESET_CNTL and ENABLE registers.

To run the counter from the card clock source the following conditions must be met:

1. ‘KILL_CLOCK = 0’ in the RESET_CNTL register.
2. ‘GPCNT_CLKSET[1:0] = 01’ in the CNTL register.

The counter will begin to count at the card clock rate as soon as these conditions are met.

To run the counter from the receive clock source the following conditions must be met:

1. ‘KILL_CLOCK = 0’ in the RESET_CNTL register.
2. ‘RCV_EN = 1’ or XMT_EN = 1 in the ENABLE register.
3. ‘GPCNT_CLKSET[1:0] = 10’ in the CNTL register.

The counter will begin to count at the receive clock rate as soon as these conditions are met.

To run the counter from the ETU (transmit) clock source the following conditions must be met:

1. ‘KILL_CLOCK = 0’ in the RESET_CNTL register.
2. Either one of the following:
   “RCV_EN = 1’ in the ENABLE register and ‘CWT_EN = 1’ in the CNTL register
   ‘XMT_EN = 1’ in the ENABLE register

3. ‘GPCNT_CLKSET[1:0] = 11’ in the CNTL register.

The counter will begin to count at the ETU clock rate as soon as these conditions are met.

The counter can be reset by setting GPCNT_CLKSEL[1–0] to 00. A 16-bit comparator value is provided that allows the software to select a count value at which an interrupt flag can be set and an interrupt generated if the mask is clear.

51.4.7 SIM LRC Block

The SIM module provides an 8-bit Linear Redundancy Check (LRC) generator/checker. The block is provided for use with T = 1 SIM cards that support LRC. This block can be enabled through the LRCEN bit in the CNTL register. This block performs an 8-bit exclusive-OR on all received or transmitted characters. At the end of the reception of a block of characters, the result is expected to be 00. If so, the LRCOK bit is set in the RCV_STAT register. During transmission, the LRC block Exclusive-ORs each character that is transmitted with the current value of the LRC. If the XMT_CRC_LRC bit in the CNTL register is set, the LRC value will automatically be sent by the SIM transmitter as the final character when the transmit FIFO empties.

The LRC value can be reset in multiple ways. Clearing the LCREN bit in the CNTL register will reset the LRC value. At the end of a transmission (either after the LRC byte is transmitted, or after the last character in the transmit FIFO is sent when XMT_CRC_LRC is clear), the LRC value is automatically reset by the SIM hardware. Finally, when setting the XMT_EN bit, the SIM hardware resets the LRC value.

51.4.8 SIM CRC Block

The SIM module provides an 16-bit Cyclic Redundancy Check (CRC) generator/checker. The block is provided for use with T=1 SIM cards that support CRC. This block can be enabled through the CRCEN bit in the CNTL register. This block performs a polynomial based check on all received or transmitted characters. The polynomial description is shown in Figure 51-50. The polynomial used is the standard CRC-CCITT where g(x) = x^16 + x^12 + x^5 + 1. The CRC register is initialized to all “1” before the data is shifted in. Before transmission the resulting CRC is inverted. For example, transmitting a 0xFA results in the following:

- Data = 0x5F (bit reversal of 0xFA)
- crc = 0x4AEA
- invert the crc = 0xB515 (bit reversal of 0xADA8
- data transmitted = 0xFA, 0xAD, 0xA8

See Figure 51-50 for illustration of the SIM CRC block.

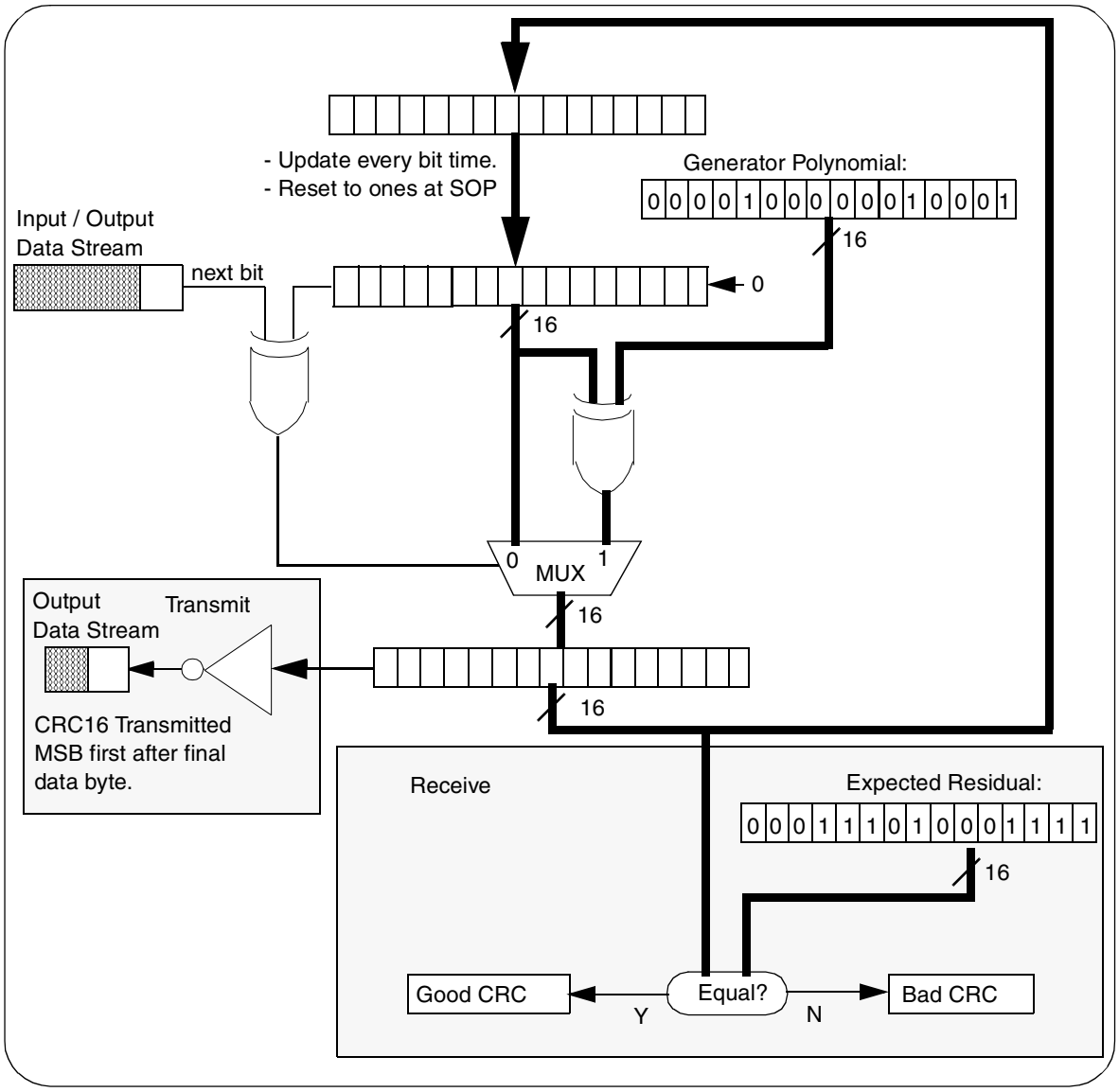


Figure 51-50. CRC Block Diagram

At the end of the reception of a block of characters, the residual from the CRC calculation is compared to 0x1D0F. If so, the CRCOK bit is set in the RCV_STAT register. During transmission, the CRC block updates the current value of the CRC residual using each character. If the XMT_CRC_LRC bit in the CNTL register is set, the CRC value will automatically be inverted and sent by the SIM transmitter as the final two characters when the transmit FIFO empties.

The CRC value can be reset in multiple ways. Clearing the CRCEN bit in the CNTL register will reset the CRC value. At the end of a transmission (either after the CRC characters are transmitted, or after the last character in the transmit FIFO is sent when XMT_CRC_LRC is clear), the CRC value is automatically reset by the SIM hardware. Finally, when setting the XMT_EN bit, the SIM hardware resets the CRC value.

51.4.9 Module Interrupts

See Table 51-39 for the list of all possible interrupt sources and their corresponding mask bits. All SIM interrupts are logically ORed to create the active low irq_n and data_irq_n signals that go to the ITC module. All mask bits are such that a logic 1 implies that the corresponding interrupt is disabled (masked), whereas a 0 implies that the interrupt enabled (unmasked). All of these mask bits are logic 1 out of reset (all interrupts are masked).

Table 51-39. SIM Module Interrupts

Flag	Flag Register	Mask	Mask Register	Description
TC	XMT_STATUS	TCIM	INT_MASK	Transmit Complete
ETC	XMT_STATUS	ETCIM	INT_MASK	Early Transmit Complete
TFE	XMT_STATUS)	TFEIM	INT_MASK	Transmit FIFO Empty
XTE	XMT_STATUS	XTM	INT_MASK	Transmit Threshold Error
TFO	XMT_STATUS	TFOM	INT_MASK	Transmit FIFO Overfill error
TDTF	XMT_STATUS	TDTFM	INT_MASK	Transmit Data Threshold Flag
GPCNT	XMT_STATUS	GPCNTM	INT_MASK	General Purpose Counter Comparator Flag
RDRF	RCV_STATUS	RIM	INT_MASK	Receive data register full (FIFO threshold level reached)
OEF	RCV_STATUS	OIM	INT_MASK	Overrun Error Flag
CWT	RCV_STATUS	CWTM	INT_MASK	Character Wait Time Counter Comparator Flag
BWT	RCV_STATUS	BWTM	INT_MASK	Block Wait Time Counter Comparator Flag
BGT	RCV_STATUS	BGTM	INT_MASK	Block Guard Time Counter Comparator Flag
RTE	RCV_STATUS	RTM	INT_MASK	Receive NACK Threshold Error
SDI1	PORT1_DETECT	SDIM1	PORT1_DETECT	SIM Detect Interrupt for port 1
SDI0	PORT0_DETECT	SDIM0	PORT0_DETECT)	SIM Detect Interrupt for port 0

51.5 Initialization/Application Information

This section describes the intended programming model for using the SIM module. The section describes how to begin a typical mode of operation using the registers.

51.5.1 Configuring SIM for Operation

The following is a list of items that need to be performed to configure the SIM module for operation:

1. To select the port in the SETUP register, do the following steps:
   a) Select which SIM module port is active by writing the SPS bit.

b) Select whether the second SIM module port is active at the same time under the control of an alternate SIM module by writing the AMODE bit.

2. Enable the selected port (for port 1, see PORT1_CNTL register; for port 0, see PORT0_CNTL) following the SIM power up procedure specified in ISO 7816.
   a) Enable power to the SIM card using the SVEN1 or SVEN0 bit.
   b) Enable SIM module transmit data output using the STEN1 or STEN0 bit. This is required to allow the SIM receiver to create NACK pulses.
   c) Enable SIM card clock using the SCEN1 or SCEN0 bit.
   d) Release the SIM card reset using the SRST1 or SRST0 bit (assert SRSTx high).
3. Select XMT port driver type (open-drain or push-pull) in the OD_CONFIG register.
   a) Configure the selected port to be open drain or push-pull by using the OD_P1 or OD_P0 bit.
4. Choose the port Baud rate in the CNTL register.
   a) Select the SIM card clock rate by using the CLK_PRESCALER[7:0] register
   b) Select the SIM card baud rate by using the BAUD_SEL[2:0] bits and SAMPLE12

NOTE

Follow the ISO 7816 spec with regards to card clock frequencies to ensure the maximum frequency specification is not violated.

5. Select Data format type, or place SIM receiver in initial character mode.
   a) Select inverse convention or direct convention by using the IC bit in the DATA_FORMAT register, or
   b) Enable initial character mode by using the ICM bit in the CNTL register

51.5.1.1 Configuring SIM Receive

The following is a list of items that need to be performed to configure the SIM receiver for operation:

1. Enable NACK capability in CNTL register.
   a) Select NACK generation on parity errors, or invalid initial character by using the ANACK bit.
   b) Select NACK generation on overrun conditions by using the ONACK bit.
2. Select the desired Receive NACK threshold and receive FIFO threshold in RCV_THRESHOLD register.
   a) Program the threshold at which the RDRF flag will be set by using the RDT bits.
   b) Program the threshold at which the RTE flag will be set by writing to the RTH[3:0] bits. If an auto power down after RTE flag is set, is desired, then enable the SIM card auto power down by setting the SAPD0,1 bits in the port0,1_cntl register.
3. Configure the Character Wait Time Counter, the Block Wait Timer Counter and the Block Guard Time Counter.
4. Enable interrupts in the INT_MASK register.
   a) Enable the receive data register full interrupt by using the RIM bit
   b) Enable the receive threshold error interrupt by using the RTM bit.

c) Enable the overrun condition interrupt by using the OIM bit
d) Enable the Character Wait Time interrupt by using the CWTM bit

51.5.1.2 Configuring SIM Transmitter

The following is a list of items that need to be performed to configure the SIM transmitter for operation:

1. Select desired re-transmission threshold for NACKed characters in XMT_THRESHOLD register.
   a) Program the threshold at which the XTE flag will be set by using the XTH[3:0] bits
2. Select the guard time between transmissions in the GUARD_CNTL register.
   a. Program the desired guard time between characters transmitted by the SIM module by using the GETU[7:0] bits
3. Select the desired Transmit FIFO threshold level in the XMT_THRESHOLD register.
   a) Program the desired threshold using the TDT[3:0] bits
4. Enable interrupts in the INT_MASK register.
   a) Enable the transmit complete interrupt by using the TCIM BIT
   b) Enable the early transmit complete interrupt by using the ETCIM bit
   c) Enable the transmit FIFO empty interrupt by using the TFEIM bit
   d) Enable the transmit threshold error interrupt by using the XTM bit
   e) Enable the transmit FIFO threshold interrupt by using the TDTFM bit
   f) Enable the transmit FIFO overfill interrupt by using the TFOM bit

51.5.1.3 Configuring SIM General Purpose Counter

The following is a list of items that need to be performed in order to configure the SIM General Purpose Counter for operation:

1. Select desired clock source for the General Purpose Counter using the CNTL register.
   a) Use the GPCNT_CLKSEL[1:0] bits to select the desired clock source for the counter
2. Program counter comparator using the GPCNT register.
   a) Use the GPCNT[15:0] bits to select the desired count value at which the GPCNT interrupt flag will be set.
3. Enable the selected clock source for the General Purpose Counter using either the RESET_CNTL or ENABLE registers.
   a) If the GP Counter is configured for the card clock, enable the clock by clearing the KILL_CLOCK bit in the RESET_CNTL register.
   b) If the GP Counter is configured for the receive oversample clock, enable this clock by setting the RCV_EN bit or XMT_EN bit in the ENABLE register.
   c) If the GP Counter is configured for the transmit oversample clock, enable this clock by setting the XMT_EN bit in the ENABLE register.
4. Enable interrupts in the INT_MASK register.
   a) Enable the general purpose counter interrupt by using the GPCNTM BIT

51.5.1.4 Configuring SIM to Measure WWT (Work Wait Time) for Type = 0 Smartcards

The following is a list of items that need to be performed in order to configure the SIM to measure work wait time. This is intended for type = 0 SmartCards. Work wait time is a combination of BWT and CWT. The CWT timer is to measure the time between the start bits of two consecutive characters in the block received from the SmartCard. If this time is larger that the value programmed in the cwt[15:0] register, then a flag will be set and an interrupt generated. This timer can be used for both type = 0 and type = 1 SmartCards. The BWT and BGT timer are used to measure the delay between the leading edge of the last character of the block received by the card and the leading edge of the first character of the next block sent by the card. If you want the SIM to enforce a WWT of 100 bits. Then, activate both the CWT and BWT and program both of them for a value of 100 bits. When measuring WWT, the BGT timer will not be used so it will be masked (inactive) by the BGTM bit. Below is the sequence to program the SIM to send and receive data while checking WWT.

1. Program both the CWT and the BWT (low and high) registers to the WWT (work wait time) value that needs to be enforced. Set BGT register to 0 (this is the default value).
2. Activate both the CWT and BWT functions by setting the CWTEN and BWTEN bit in the CNTL register.
3. Enable the CWT and BWT interrupts by clearing the CWTM and BWTM bits in the INT_MASK register.
4. Program the data to send to the SmartCard by writing data to XMT_BUFFER
5. Activate the SIM transmit and receive by setting bits XMT_EN and RCV_EN in the ENABLE register.
6. If the software has more data to send, then as the FIFO starts to get near empty, it should write more data to the XMT_BUFFER. If the software does not have any more data to send then proceed to the next step.
7. When SmartCard has completed its transmission to the SIM module, disable the BWT by clearing the bit BWTEN in the CNTL register. This will prepare SIM to measure the next block wait time.
8. Disable the SIM transmitter by clearing the XMT_EN bit in the ENABLE register.
9. Program the next data to send by writing data to the XMT_BUFFER.
10. Enable the BWT function by setting the BWTEN bit in the CNTL register.
11. Enable the SIM transmitter by setting the XMT_EN bit in the ENABLE register.
12. Steps 6 to 10 can be repeated for each transmission to the SmartCard.
13. If a CWT or a BWT interrupt occurs, then the software should consider that a WWT violation. The software should clear the CWT or BWT interrupt by writing a “1” to the BWT or CWT bit in the RCV_STATUS register.

51.5.1.5 Configuring SIM to measure CWT, BWT, BGT for type=1 SmartCards

The following is a list of items that need to be performed in order to configure the SIM to measure CWT, BWT, BGT. This is intended for type = 1 SmartCards. The CWT timer is to measure the time between start bits in the consecutive characters received from the SmartCard. If this time is larger that the value programmed in the CWT[15:0] register, then a flag will be set and an interrupt generated. This timer can

be used for both type = 0 and type = 1 SmartCards. The BWT, BGT timer is used to measure the delay between the leading edge of the last character of the block received by the card and the leading edge of the first character of the next block sent by the card. If this time is larger than the value in the both BWT registers (BWT and BWT_H), then the BWT flag will be set and an interrupt generated. If the time is less than the value in the BGT register, then the BGT flag will be set and an interrupt generated.

1. Program the CWT, BWT, BWT_H and BGT registers to the value that needs to be enforced.
2. Activate both the CWT and BWT functions by setting the CWTEN and BWTEN bit in the CNTL register.
3. Enable the CWT, BWT, BGT interrupts by clearing the CWTM, BWTM, BGTM bits in the INT_MASK register.
4. Program the data to send to the SmartCard by writing data to XMT_BUFFER
5. Activate the SIM transmit and receive by setting bits XMT_EN and RCV_EN in the ENABLE register.
6. If the software has more data to send, then as the FIFO starts to get near empty, it should write more data to the XMT_BUFFER. If the software does not have any more data to send then proceed to the next step.
7. When SmartCard has completed its transmission to the SIM module, disable the BWT by clearing the bit BWTEN in the CNTL register. This will prepare SIM to measure the next block wait time.
8. Disable the SIM transmitter by clearing the XMT_EN bit in the ENABLE register.
9. Program the next data to send by writing data to the XMT_BUFFER.
10. Enable the BWT function by setting the BWTEN bit in the CNTL register.
11. Enable the SIM transmitter by setting the XMT_EN bit in the ENABLE register.
12. Steps 6 to 10 can be repeated for each transmission to the SmartCard.
13. If a CWT or a BWT interrupt occurs, then the software should read the RCV_STATUS register to determine if the error was caused by a CWT, BWT, or a BGT violation. The software should clear the CWT, BWT, BGT interrupt by writing a “1” to the CWT, BWT, BGT bit in the RCV_STATUS register.

51.5.1.6 Configuring SIM Linear Redundancy Check (LRC) Block

The following is a list of items that need to be performed in order to configure the SIM Linear Redundancy Check Block for operation:

1. Enable the LRC block by using the CNTL register
   a) Use the LRCEN bit to enable the LRC block
   b) Use the XMT_CRC_LRC bit to enable the transmission of the LRC Character after the last character in the Transmit FIFO is sent. Refer to the T=1 programming model for more details.

51.5.1.7 Configuring SIM Cyclic Redundancy Check (CRC) Block

The following is a list of items that must be performed to configure the SIM Cyclic Redundancy Check Block for operation:

1. Enable the CRC block by using the CNTL register.
   a) Use the CRCEN bit to enable the CRC block
   b) Use the XMT_CRC_LRC bit to enable the transmission of the CRC Characters after the last character in the Transmit FIFO is sent. Refer to the T=1 Programming model for more details.

51.5.2 Using the SIM Receiver

Once the SIM has been properly configured (such as correct baud rate and correct data format), SIM receptions can be enabled by setting the receive enable bit, RCV_EN, in the ENABLE register. As bytes are received, they will be placed in the 288 byte deep receive data FIFO. Unread bytes can be accessed from this FIFO at any time. There is no need to disable the receiver to access the FIFO. The FIFO should only be read when the receive FIFO data flag, RFD, in the RCV_STATUS register is set. The RFD flag, which cannot create an interrupt, is high any time there is at least one unread byte in the receive FIFO. If the receive FIFO is read when RFD is low, it will simply produce the last byte read.

The receive data register full flag, RDRF, in the RCV_STATUS register can be used to determine when the receive FIFO has reached a given threshold value. This flag will create an interrupt if the RIM bit in the INT_MASK register is clear. To control at which point RDRF is set, program the receive data threshold, RDT, in the RCV_THRESHOLD register. If the number of unread bytes in the receive FIFO is equal to or greater than the value set by RDT, RDRF will be set.

NOTE

A value of 0x0 in RDT implies that there must be 288 unread bytes in the receive FIFO to trigger RDRF.

The value in RDT can be changed at any time to alter this threshold level. The comparison between the number of unread bytes in the FIFO and the value set by RDT is continuously updated so that any change in either will be immediately reflected in the state of RDRF. For instance, if RDT is set to 5, and there are 3 unread bytes in the FIFO, changing RDT to 2 will immediately cause RDRF to be set. Likewise, setting RDT back to 5 will cause RDRF to clear. Similarly, if there are 5 unread bytes in the receive FIFO and RDT is set to 3, RDRF will remain high until 3 reads are complete, assuming RDT is left as a constant and no new data is received.

The standard flow for receiving bytes from the SIM card is to set RDT to the appropriate value, wait for RDRF to cause an interrupt (RIM clear), and then read bytes out of the receive FIFO as long as RFD is high. In addition to checking RFD between every byte, it is also recommended that software check for the existence of a set OEF flag as well.

51.5.2.1 Receive Parity Errors and Parity NACK Generation

The SIM receiver checks every byte received for proper parity. The IC control bit in the DATA_FORMAT register controls whether it checks for odd parity or even parity. When checking for odd parity, the number of logic ones contained in the 9 received bits (8 data bits and 1 parity bit) should be odd. Likewise, when

checking for even parity, the number of logic ones contained in the 9 received bits (8 data bits and 1 parity bit) should be even.

When a parity error is detected on a given byte, the PORTx_PE bit for that byte is set in the FIFO. The PORTx_PE flag for each byte is read out of the FIFO when the data itself is read. There is no need to attempt to clear the parity error flag in the FIFO. It is simply overwritten the next time a byte is received into that position of the FIFO. A parity error cannot cause an interrupt.

It is possible for the SIM to automatically request that the SIM card re-send a byte found to have a parity error by generating a NACK pulse on the SIM XMT pin. The SIM will generate a NACK pulse on a byte received with a parity error if the control bit ANACK in the CNTL register is set. Bytes with parity errors that cause a NACK pulse are still placed into the FIFO just as bytes that do not cause a NACK pulse. It is up to the software to discard data bytes with parity errors.

To control NACK generation by the SIM receiver use the RTH[3:0], Receive NACK threshold, in the RCV_THRESHOLD register. This set of bits specify the number of consecutive NACK's generated by the SIM module on a received byte, before generating an RTE (receive threshold interrupt flag) in the RCV_STATUS register. The RTE flag would also force the SIM port to power down the card if the SAPD0,1 bit is set in the PORT0,1_CNTL register.

NOTE

The ANACK bit must be set in the CNTL register to enable this feature. The control bit ANACK is also used in initial character mode to enable the retransmission of initial characters in the event that an invalid initial character is received.

When a valid character has been received by the SIM, the internal counter keeping track of the number of NACK's transmitted on the current byte resets to zero. Clearing the receive threshold error (RTE) bit would also clear that counter.

When generating a NACK pulse, the SIM will generate the low pulse starting at 10.5 ETUs and lasting for 1 ETU.

51.5.2.2 Receive Frame Errors

The SIM receiver checks every byte received for a proper stop bit. A stop bit should exist during at least the first half of the 11th ETU time after the start of the character. If this is not true, a frame error is flagged. When a frame error is detected on a given byte, the PORTx_FE bit for that byte is set in the FIFO. The PORTx_FE flag for each byte is read out of the FIFO when the data itself is read. There is no need to clear the frame error flag in the FIFO. It is simply overwritten the next time a byte is received into that position of the FIFO. A frame error cannot cause an interrupt, nor can it create a NACK pulse to the receiver asking for a retransmission of the corrupted data.

51.5.2.3 Receive Overrun Errors and Overrun NACK Generation

When there already exists 288 unread bytes in the FIFO, a received character will cause the SIM receiver to flag an overrun condition. This condition will always set the overrun error flag, OEF in the

RCV_STATUS register. The received byte will be discarded leaving the 288 unread bytes in the FIFO unaltered.

It is possible for the SIM to automatically request that the SIM card resend the byte that caused the overrun condition by generating a NACK pulse on the SIM XMT pin. The SIM will generate a NACK pulse for the byte that caused the overrun condition if the control bit ONACK in the CNTL register is set. In this case, the existence of an OEF flag does not indicate the loss of data, but rather a NACK (that is, retransmission request) due to a full receive FIFO. As opposed to transmit NACK generation, there is no limit to the number of times an overrun condition will cause a NACK other than to disable ONACK itself. When generating a NACK pulse, the SIM will generate the low pulse starting at 10.5 ETUs and lasting for 1 ETU.

If the control bit ONACK is clear, the existence of a high OEF flag indicates the loss of data. The OEF flag can create an interrupt if the OIM bit in the INT_MASK register is clear.

To clear the OEF flag, software must simply write a “one” to the OEF bit position in the RCV_STATUS register. A high OEF flag has no effect on the operation of the SIM receiver other than to create an interrupt if OIM is clear.

51.5.2.4 Using Initial Character Mode and Resulting Receive Data Formats

The SIM receiver supports the detection of special characters that allow it to determine what data format is being used by the connected SIM card. When placed in initial character mode, the SIM expects to receive one of two potential values that it will use to set the data format control bit, IC, in the DATA_FORMAT register.

The two possible data formats are inverse convention and direct convention. Essentially, inverse convention differs from direct convention in that the order of the data is flipped msb for lsb, and the data bits and parity bit are logically inverted. When receiving inverse convention data, the transformation of the data back to direct convention format is done in hardware, including the inversion of the data and parity bits.

To place the SIM into initial character mode, set the ICM bit in the CNTL register. Once a valid initial character is received, the IC bit in the DATA_FORMAT register will be set accordingly by the hardware, and the ICM bit will be cleared. Software can read the state of the IC bit to determine which mode the SIM is currently using.

The 0x3B (as decoded by direct convention) with parity bit high will cause direct convention to be used (IC set to logic 0), whereas a 0x3F (as decoded by inverse convention) with parity bit high will cause inverse convention to be used (IC set to logic 1).

When the receiver is in initial character mode, all received bytes will continue to be placed into the receive FIFO whether they be valid initial characters or not. If a valid initial character is received that causes the data format being used to change, all subsequent bytes will be decoded with that format before being placed into the FIFO, including the initial character byte itself. That is, if the IC bit is low, and the correct initial character for setting inverse convention is received, that character and all subsequently received characters will be stored in the FIFO after having been decoded using inverse convention (for example, the initial character will be stored as 0x3F).

If the receiver is in initial character mode (ICM is high), and an invalid initial character is received, the SIM can be configured to automatically request that the initial character be retransmitted by setting the ANACK control bit in the CNTL register. When generating a NACK pulse, the SIM will generate the low pulse starting at 10.5 ETUs and lasting for 1 ETUs. The invalid initial character will be placed into the receive FIFO and marked with a parity error to signify that this is an invalid initial character.

51.5.2.5 Initial Character Mode Programming Notes

The usage of the initial character mode requires close attention to the programming model. There is a condition where a parity error in the initial byte for direct convention (0x3B) could be decoded as what appears to be a valid initial character for inverse convention (that is, 0x3F). The SIM module will not recognize this as a valid initial character for inverse convention and will mark the character by setting the parity error flag. The software must look for the existence of a parity error before recognizing a character as a valid initial character.

51.5.2.6 Automatic Receiver Mode

The SIM module has an automatic receive mode that inhibits the data being transmitted by the SIM module from entering the SIM receive buffer through the feedback path of the SIM data pin. The SIM module receiver should normally be enabled while the transmitter is operational. Automatic receive mode saves the software from having to actively manage the transition from transmitter to receiver. The auto receive mode is always active whenever the receiver is enabled.

51.5.2.7 Using the SIM Receiver with “T=1” SIM Cards

The SIM module provides hardware support for “T=1” type SIM cards. These type of cards present several requirements above and beyond the standard “T=0” cards. The features provided to meet the requirements that pertain to the SIM receiver are as follows:

- 11 ETU Characters
  - “T=1” cards can transmit with character lengths of 11 ETUs (that is, one STOP bit). The SIM module provides the RCVR11 bit in the GUARD_CNTL register in order to configure the receiver state machine to accept 11 ETU characters.
- Character Wait Time Counter
  - The character waiting time (CWT) is defined as the time between the start bits of two consecutive characters received from the SmartCard. The value of CWT can range from 12 ETU to 32779 ETU. The SIM module provides a 16-bit counter with programmable comparator clocked at the ETU bit rate to identify when the CWT has been exceeded by the SIM card.
- Block Waiting Time
  - The block waiting time (BWT) is defined as the maximum time between the start bits of the last character of a transmitted block and the first character of the next received block. The block wait time must not exceed a value that is programmable. The SIM module provides a 32-bit Block Wait Timer that can be used to identify when the BWT has been violated (divided in two registers).

- Block Guard Time
  - The block guard time (BGT) is defined as the minimum delay between the start bits of the last character of a transmitted block and the first character of the next received block. The block guard must be greater than a value that is programmable. The SIM module provides a 16-bit Block Guard Timer that can be used to identify when the BGT has been violated.
- Error Detection Code
  - "T=1" cards can specify LRC or CRC error detection codes to be used. The SIM module provides hardware support for both the LRC and CRC operations.

51.5.3 Using SIM Transmitter

Once the SIM has been properly configured (such as data XMT enabled, correct baud rate, and correct data format), the transmitter can be enabled by setting the XMT_EN bit in the ENABLE register. If data had been previously written to the transmit FIFO, the transmitter will begin to send the first character. If no data is written to the transmit FIFO before enabling the transmitter, then the transmitter will wait until the first character is written before beginning transmission. Clearing the XMT_EN bit while the transmitter is in operation, will halt any transmission in progress, flush the transmit FIFO, and reset the transmit state machine.

Data can be written to the transmit FIFO at any time.

The transmit data threshold flag, TDTF, in the XMT_STATUS register can be used to determine when the transmit FIFO has reached a given threshold value. This flag will create an interrupt if the TDTFM bit in the INT_MASK register is clear. To control at which point TDTF is set, program the transmit data threshold, TDT[3:0], in the XMT_THRESHOLD register. If the number of bytes remaining in the transmit FIFO is equal to or less than the value set by TDT[4:0], TDTF will be set.

NOTE

A value of 0x0 in TDT[3:0] implies that the transmit FIFO must be empty to trigger TDTF.

The value in TDT[3:0] can be changed at any time to alter this threshold level. The comparison between the number of remaining bytes in the transmit FIFO and the value set by TDT[3:0] is continuously updated so that any change in either will be immediately reflected in the state of TDTF. Unlike the RDRF flag for the receive FIFO, TDTF is latched and remains set until the software writes a 1 to the TDTF bit location in the XMT_STATUS register. For instance, if TDT[3:0] is set to 5, and there are 6 bytes remaining in the FIFO, changing TDT[3:0] to 6 will immediately cause TDTF to be set. However, setting TDT[3:0] back to 5 will not cause TDTF to clear.

The standard flow for transmitting bytes from the SIM card is to set TDT[3:0] to the appropriate value, write up to 16 bytes to the transmit FIFO, enable the transmitter, wait for TDTF to cause an interrupt (TDTFM clear), and then write additional bytes to the transmit FIFO.

NOTE

For the SIM module to transmit successfully, the transmit pin must be connected to the receive pin of the same port. This connection is necessary for the SIM to decode transmit NACKs sent to it by the SIM card. Without this connection, all transmissions will appear to have a NACK thereby causing the first byte to be transmitted continuously. When operating in external one wire interface mode (3VOLT0 or 3VOLT1), this connection is made internal to the SoC.

51.5.3.1 Transmit Data Formats

There are two possible data formats that the SIM module uses when transferring data to the SIM: card — inverse convention or direct convention. The format used depends on the state of inverse convention bit (IC in the DATA_FORMAT register). Software can set the IC bit, or it can be set automatically by hardware when using the initial character mode.

51.5.3.2 Transmit NACK

The SIM transmitter can respond to NACKs created by the SIM card. A NACK will be decoded if the SIM card creates a logic low level on the SIM receive pin during the STOP bit time at the end of a transmitted byte. To prevent a situation where the SIM interface is stalled by an infinite number of NACK pulses on a given byte, the SIM module can be configured to limit the number of times it will respond to NACKs. The XTH[3:0] control field in the XMT_THRESHOLD register allows software to set a threshold on the number of times a given byte will be retransmitted. If the threshold is reached, a transmit threshold error, XTE in the XMT_STATUS register, is asserted.

When XTE is set, the SIM transmitter is halted, and all pending transfers are aborted, and the TC, ETC, AND TFE flags are set. All bytes remaining in the transmit FIFO are lost. There is no way to restart the transmission on the next byte in the FIFO. The transmitter remains frozen until XTE is cleared by software. The only way to clear XTE is to write a 1 to the XTE bit in the XMT_THRESHOLD register. The XTE flag can create an interrupt if the XTM mask in the XMT_THRESHOLD register is clear.

It is possible to disable the detection of NACKs from the SIM card by setting XTH[3:0] to 0x0. By setting XTH[3:0] to 0x1, it is possible to disable all retransmissions while still setting XTE on the first NACK received. In general, XTE is set on the NACK that causes the threshold set by XTH[3:0] to be reached. This final NACK does not cause a retransmission whereas all previous NACKs will cause a retransmission.

51.5.3.3 Transmit Guard Time

The time between data bytes sent from the SIM transmitter can be altered using the transmit guard time control. By default, the minimum time between start bits of successive transmitted bytes is 12 ETUs (Elementary Time Units). An ETU is equivalent in time to 1 bit time. Therefore, the amount of time an

ETU consumes is determined by the baud rate chosen. The 12 ETUs that form the default character transmission time consist of a start bit, 8 data bits, a parity bit and two stop bits. The number of stop bits (idle bits) can be extended by an integer number of ETUs. The number of additional ETUs can be programmed directly into the GETU[7:0] bits of the GUARD_CNTL register. Programming the GETU[7:0] bits to 0xFF will configure the SIM transmitter to use only one stop bit for each character transmission.

51.5.3.4 Using SIM Transmit with “T=1” SIM Cards

The SIM module provides hardware support for “T=1” type SIM cards. These type of cards present several requirements above and beyond the standard “T=0” cards. The features provided to meet the requirements that pertain to the SIM transmitter are as follows:

- 11 ETU Characters
  - The SIM module transmitter has a programmable guard time register that allows the programmer to specify the number of ETUs between character transmissions. Programming a value of 255 (0xFF) in the GETU[7:0] bits in the GUARD_CNTL register will set the number of ETUs per character transmitted to 11.
- Character Waiting Time
  - The character waiting time (CWT) is defined as the time between the start bits of two consecutive characters. The value of CWT can range from 12 ETU to 32779 ETU. The time between transmitted characters is controlled by the programmable guard time in the GUARD_CNTL register. However, the time between the last byte in the transmit FIFO, and the next transmitted byte can be largely affected by software response time to the transmit interrupts. The SIM transmitter provides a Transmit FIFO threshold (TDTF) interrupt to signal the system when the expected number of characters have been transmitted from the transmit FIFO. The minimum CWT is achieved only if the software can respond to the TDTF interrupt and write new data to the transmit FIFO before the last character in the Transmit FIFO has been sent.
- Block Waiting Time
  - The block waiting time (BWT) is defined as the maximum time between the start bits of the last character of a transmitted block and the first character of the next received block. The value of BWT is always greater than 1800 ETU. The SIM transmitter provides a General Purpose Counter that can be used to track the BWT. The BWT is purely determined by software response time to the transmit interrupts.
- Block Guard Time
  - The block guard time (BGT) is defined as the minimum delay between the start bits of the last character of a transmitted block and the first character of the next received block. The value of BGT is 22 ETU. The SIM module supports the BGT by providing the ability to generate an interrupt when the last byte is received, and transmitting within two ETU after the XMT_EN bit is set. The BGT will be determined by the speed at which the software can react to an interrupt and enable the transmitter.

- Error Detection Code
  - “T=1” cards can specify LRC or CRC error detection codes to be used. The SIM module provides hardware support for both the LRC and CRC operation.

51.5.4 Suggested “T=1” Compliant Programming Model

This section describes the suggested programming model for supporting “T=1”, “T=0”, and known “special” cards using the SIM module on the SoC. This should be used as a rough guide for how to configure the SIM module for use with SIM cards specified by ISO 7816-3 and EMV. Some details are not addressed. Other uses for some of the SIM features are not included (for example, GP Counter uses for some ISO timing requirements).

51.5.4.1 Answer To Reset (ATR) Detection

The first step to communicating with a SIM card is to provide power and a clock signal to the card. Once the card is detected as present (using the presence detect features, or some other method), the SIM card should be powered up according to the power up sequence specified in the ISO 7816-3 specification.

1. Apply voltage to the SIM card by setting the SVEN0 or SVEN1 bit in the PORTx_CNTL register
2. Select the appropriate clock frequency for the SIM card by programming the CLK_PRESCALER[7:0] register.
3. Enable the clock to the SIM card by setting the SCEN0 or SCEN1 bit in the PORTx_CNTL register
4. Remove the card from reset by setting the SRST0 or SRST1 bit in the PORTx_CNTL register

The first communication between the SIM card and the SIM module will be a block of data sent from the SIM card to the SIM module after the card is powered and the card reset is removed. This block is called the Answer To Reset (ATR). To receive the ATR, the SIM module should be configured for 12 ETU character reception. According to the ISO 7816-3 spec, both “T=0” and “T=1” cards will communicate initially using 12 ETU character durations.

NOTE

We are aware of some card manufacturers that communicate at 11.5 ETU character durations (Geldkarte). This will complicate the initial card detection sequence shown below.

5. Clear RCVR11 bit in the GUARD_CNTL register

The next item to configure for ATR reception is the NACK capability of the SIM module. The ISO 7816-3 spec allows the SIM module to NACK any communication errors that occur during the initial communication at 12 ETU.

NOTE

The Europay Mastercard and VISA (EMV) cards are similar to “T=1”, but do not allow the SIM module to NACK during the initial communication. This will again complicate the initial card detection sequence shown below.

6. Set ANACK bit in the CNTL register to enable NACK generation

In order for the SIM module to notify when characters are received, the receive interrupt(s) can be enabled. A threshold can be set for the number of characters to receive before generating an interrupt.

7. Enable RDRF and OEF interrupts by setting RIM and OIM bits in INT_MASK register
8. Set desired threshold for received characters by writing RDT in RCV_THRESHOLD register

The SIM should be setup to perform in initial character mode. This will cause the hardware to identify the first valid character sent during the ATR as an initial character. This character will automatically configure the hardware for the data convention used by the SIM card.

9. Set Initial Character Mode by setting the ICM bit in the CNTL register

The ISO 7816-3 spec requires that SIM cards meet certain timing restrictions. One of these is the time from the deassertion of the card reset to the beginning of the ATR sequence. The SIM module General Purpose Counter can be used to verify that the SIM card begins its ATR within the 400 to 40,000 clock cycle range.

1. Set General Purpose Counter Comparator to 0x9C40 using GPCNT register.
2. Enable General Purpose Counter Interrupt by clearing GPCNTM in INT_MASK register.
3. Enable General Purpose Counter by programming the GPCNT_CLKSEL[1:0] bits to 01 so the card clock is used for counting.

The ISO7816-3 spec states that the maximum allowed time between two characters during the ATR is 9600 ETUs (Initial Waiting Time). The Character Wait Time Counter should be setup to detect any errors for this condition.

1. Set Character Wait Time Counter Comparator to 9600 using the CHAR_WAIT register
2. Enable the Character Wait Time Counter Interrupt by clearing CWTM in INT_MASK register
3. Enable the Character Wait Time Counter by setting the CWTEN bit in the CNTL register

The last step in preparing for ATR reception is to enable the receiver.

4. Set RCV_EN bit in ENABLE register

The SIM module will generate interrupts once a threshold number of characters is received. The software should react to these interrupts and read the characters from the receive FIFO (RCV_BUF) until the complete ATR has been received. If a General Purpose Counter interrupt occurs before the final ATR character is received, then the card should be deactivated according to the ISO 7816-3 spec. Otherwise, once a valid ATR is received, the software will know from the ATR information the specific characteristics for this card (refer to the ISO 7816-3 spec for details).

51.5.4.2 Programming Considerations for Geldkarte Cards

There is at least one manufacturer of SIM cards (Geldkarte) that we know of that does not send the ATR in 12 ETU mode or support NACKs. This creates an issue with detecting a valid ATR from a SIM card. Since any kind of card can be attached to the SIM module, the software and hardware do not know how to begin communication. Basically, if the card fails to send a valid ATR on the first try, disable NACKs, configure the SIM module for 11 ETU and try again. The software should toggle between 12 and 11 ETU modes with and without NACKs enabled until a valid ATR is received, or the number of attempts to communicate passes a predetermined error threshold.

Figure 51-51 shows the flow chart for the suggested Geldkarte Compliant SIM Initialization.

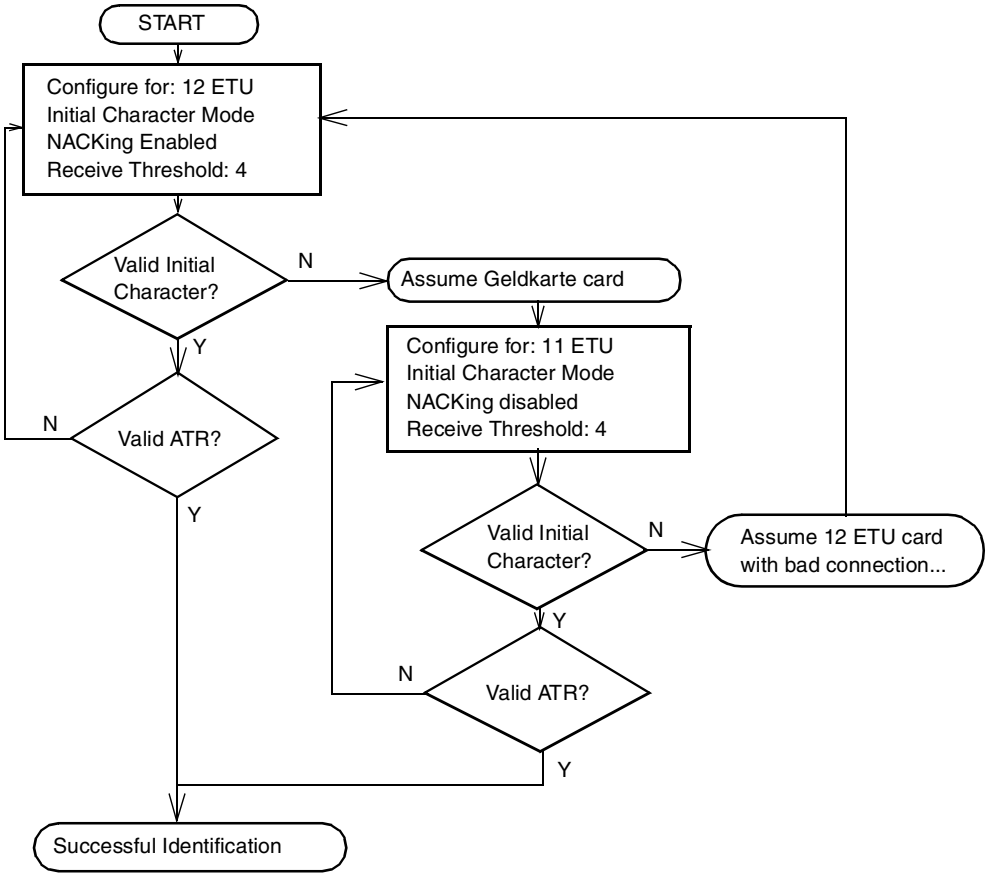


Figure 51-51. Suggested "T=1", EMV, Geldkarte Compliant SIM Initialization

51.5.4.3 Programming Considerations for T=0 SIM Cards

If the card is of type T=0, the software should adjust the following parameters according to the information in the ATR:

1. Adjust the baud rate by changing the values of BAUD_SEL[2:0] and SAMPLE12 in the CNTL register
2. Adjust the guard time between characters by changing the value of GETU[7:0] in the GUARD_CNTL register
3. Adjust NACK capability by modifying the values of the ONACK and ANACK bits in the CNTL register
4. Adjust the stop clock polarity by modifying the values of the SCSP0 or SCSP1 bits in the PORTx_CNTL register
5. Adjust the level of transmit NACK re-transmissions allowed by modifying the value of the XTH[3:0] bits in the XMT_THRESHOLD register
6. Adjust the level for the Receive NACK threshold by modifying the RTH[3:0] bits in the RCV_THRESHOLD register.

If a negotiation with the SIM card is desired, the software sends a PPS response to the SIM card. To send the response, the following steps should be performed:

1. Set the desired transmit FIFO threshold level by writing the TDT[3:0] bits in the XMT_THRESHOLD register
2. Write the characters to be sent as response (max 16) to the transmit FIFO using the XMT_BUF register
3. Clear all transmit interrupt flags in the XMT_STAT register by writing a one to them
4. Enable the transmit interrupts desired by clearing the mask bits in the INT_MASK register. If more than 16 character are to be sent, it is suggested that the TDTF interrupt be used to signify when to write more characters to the transmit FIFO. This results in the most efficient transfer times to the SIM card.
5. Enable the transmitter by setting the XMT_EN bit in the ENABLE register

At this point, the SIM module will transmit the characters in the transmit FIFO. If more than 16 characters are to be sent, the transmit threshold interrupt will be set when the threshold number of characters are remaining in the FIFO. The software can then write an additional number of characters to be sent without interrupting transmission to the SIM card.

Once the transmission is complete, the SIM module should be completely configured for standard operation with the T=0 SIM card. The software can continue to service RDRF interrupts for received characters, and TDTF interrupts for transmitted characters.

51.5.4.4 Programming Considerations for T=1 SIM Cards

If the card is of type T=1, the software should adjust the following parameters according to the information in the ATR:

1. Adjust the baud rate by changing the values of BAUD_SEL[2:0] and SAMPLE12 in the CNTL register
2. Adjust the guard time between characters by changing the value of GETU[7:0] in the GUARD_CNTL register. Setting GETU[7:0] to 0xFF configures the SIM transmitter for 11 ETU transmissions
3. Disable NACK capability by clearing the ONACK and ANACK bits in the CNTL register. T=1 cards do not allow NACKs.
4. Adjust the stop clock polarity by modifying the values of the SCSP0 or SCSP1 bits in the PORTx_CNTL register
5. Set Character Wait Time Counter Comparator to value specified in the ATR by using the CHAR_WAIT register
6. Enable the Character Wait Time Counter Interrupt by clearing CWTM in INT_MASK register
7. Enable the Character Wait Time Counter by setting the CWTEN bit in the CNTL register
8. Enable CRC or LRC error checking according to the ATR information by setting either the CRCEN or LRCEN bit in the CNTL register. These bits will never be set at the same time!

For T=1 cards, the ATR is sent using a T=0 type of structure (12 ETU, no LRC or CRC). If a negotiation with the SIM card is desired, the software will send a PPS response to the SIM card. Otherwise, the

protocol is initiated with a block transfer from the SIM module. In order to send the response or the first block, the following steps should be performed:

9. Set the desired transmit FIFO threshold level by writing the TDT[3:0] bits in the XMT_THRESHOLD register
10. Write the characters to be sent as response (max 16) to the transmit FIFO using the XMT_BUF register
11. Clear all transmit interrupt flags in the XMT_STAT register by writing a 1 to them
12. Enable the transmit interrupts desired by clearing the mask bits in the INT_MASK register. If more than 16 character are to be sent, it is suggested that the TDTF interrupt be used to signify when to write more characters to the transmit FIFO. This results in the most efficient transfer times to the SIM card.
13. Enable the transmission of the error checking characters (LRC or CRC) by setting the XMT_CRC_LRC bit in the CNTL register.

NOTE

If the card supports PPS, the software may not be allowed to send the LRC/CRC information until the PPS exchange is completed. If so, do not set the XMT_CRC_LRC bit during the PPS exchange.

14. Enable the transmitter by setting the XMT_EN bit in the ENABLE register

At this point, the SIM module will transmit the characters in the transmit FIFO. If more than 16 characters are to be sent, the transmit threshold interrupt will be set when the threshold number of characters are remaining in the FIFO. The software can then write an additional number of characters to be sent without interrupting transmission to the SIM card.

Once the transmission is complete, the SIM module should be completely configured for standard operation with the T=1 SIM card. The software can continue to service RDRF interrupts for received characters, and TDTF interrupts for transmitted characters.

Chapter 52
Smart Direct Memory Access (SDMA) Controller

52.1 Introduction

The Smart Direct Memory Access (SDMA) controller offers highly-competitive DMA features combined with software-based virtual-DMA flexibility. It enables data transfers between peripheral I/O devices and internal/external memories.

The SDMA controller helps maximize system performance by off-loading the CPU in dynamic data routing.

52.1.1 Overview

Figure 52-1 shows a block diagram of the SDMA controller. It includes the custom RISC core along with its RAM, ROM, DMA units, and the scheduler.

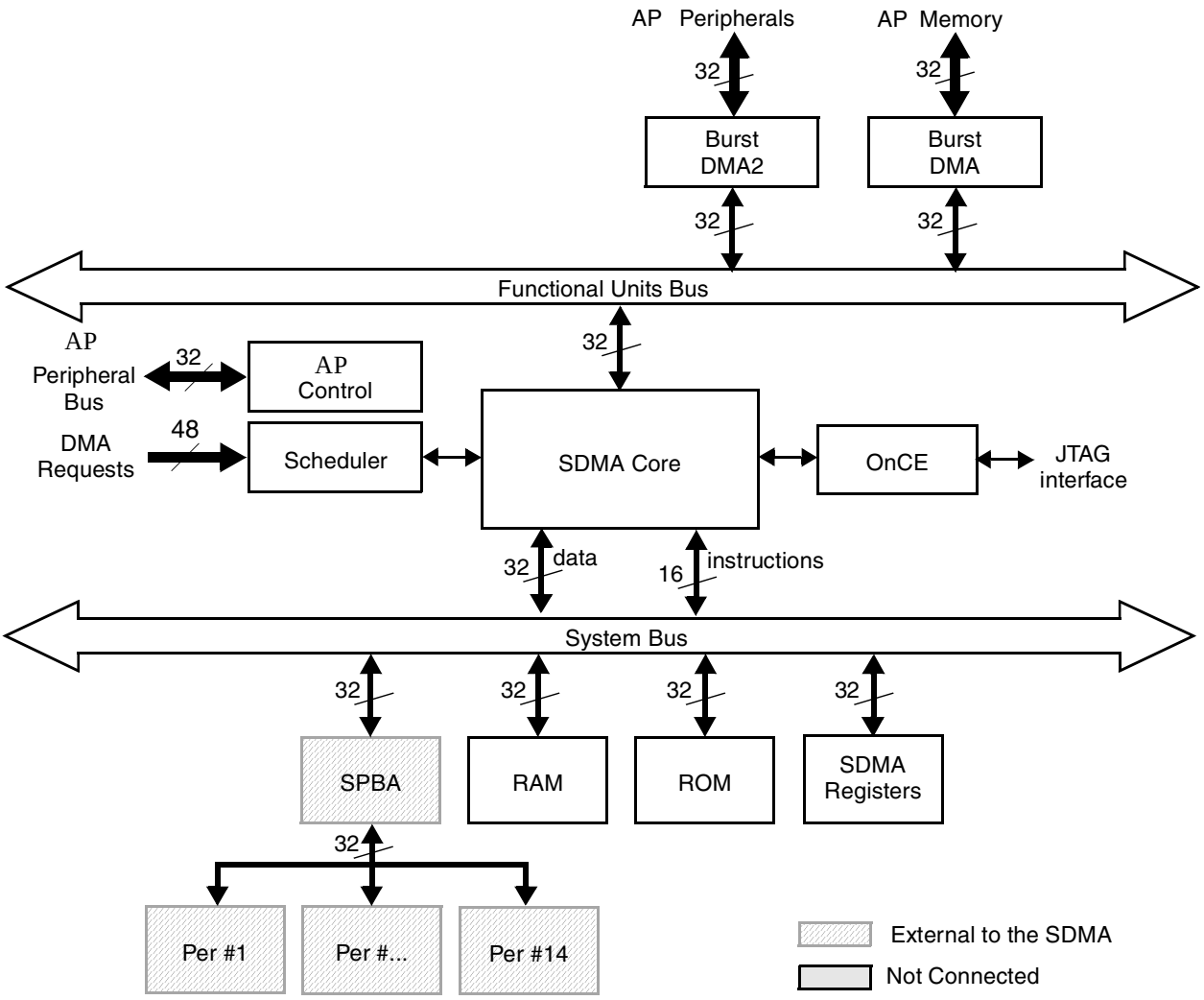


Figure 52-1. SDMA Block Diagram

The SDMA core executes short routines that perform DMA transfers; these routines are called *scripts*. The SDMA core interfaces to its own memory via the SDMA system bus. The SDMA system bus supports a 32-bit data path and a 16-bit address bus. The system bus datapath is used for both 16-bit instruction (program) memory access and 32-bit data access. DMA units interface to the core via the Functional Unit Bus and use dedicated registers to perform DMA transfers.

The SDMA memory contains a ROM and a RAM. The ROM contains startup scripts (for example, boot code) and other common utilities, which are referenced by the scripts that reside in the RAM. The internal RAM is divided into a context area and a script area (more details about this mapping are available in Section 52.13.5.1, Instruction Memory Map" and Section 52.13.5.2, Data Memory Map").

Every transfer channel requires one context area to keep the contents of all the core and unit registers while inactive. Channel scripts are downloaded into the internal RAM by the SDMA using a dedicated channel that is started during the boot sequence. Downloads are invoked using commands and pointers provided by the AP. Every channel contains a corresponding channel script located in RAM and/or ROM that can

be reconfigured independently as-needed. Channel scripts can be stored in an external memory and downloaded when needed. The SDMA can be configured with any mixture of scripts to enable an endless combination of supported services.

The scheduler monitors and detects DMA requests, mapping them to channels, and mapping individual channels to a pre-configured priority. At any given point, the scheduler presents the highest priority channel that requires service to the SDMA core. A special SDMA core instruction is used to “conditionally yield” the current channel being executed to an eligible channel that requires service. If (and only if) there is an eligible channel pending, will the current channel execution be preempted.

There are two `yield` instructions that differently determine the eligible channels: In the first version, eligible channels are pending channels with a strictly higher priority than the current channel priority. In the second version (`yieldge`), eligible channels are pending channels with a priority that is greater or equal to the current channel priority. The scheduler detects devices that need service through its 48 DMA request inputs. After a request is detected, the scheduler determines the channel(s) that is (are) triggered by this request and marks it (them) as pending in the “Channel Pending (EP)” register. The priorities of all the pending channels are continuously evaluated in order to update the highest pending priority. The channel pending flag is cleared by the channel script when the transfer has completed.

The AP Control module contains the control registers used to configure the 32 individual channels. There are 48 Channel Enable registers, and every register maps one DMA request to any desired combination of channels. The 32 Priority registers are used to assign a programmable 1-of-7 level priority to every possible channel. This module also contains all other control registers that the AP can access.

The 48 DMA requests that are connected to the scheduler come from a variety of sources. The “receive register full” and “transmit register empty” signals found in the UART and USB ports are typical examples of DMA requests that can be connected to the SDMA. These requests can be used to trigger a specific SDMA channel, or several channels.

There is an OnCE compatible debug port for product development. The OnCE includes support for setting breakpoints, single-step and trace, and register dump capability. In addition, all memory locations are accessible from the debug port.

52.1.2 Features

The following are the SDMA features:

- Multi-channel DMA supporting up to 32 time-division multiplexed DMA channels
- Hardware or software driven triggers for each channel
- 48 hardware driven triggers that can be mapped to any channel.
- Memory accesses including linear addressing, FIFO addressing and 2D addressing
- Fast context-switching with two-level, priority-based preemptive multi-tasking
- 16-bit instruction-set micro-RISC engine (the SDMA core)
- Two DMA units with some or all the following features:
  - Auto-flush and prefetch capability
  - Flexible address management (increment, decrement, and no address changes on source and destination address)

- Misaligned data-transfer support
- Uni-directional and bi-directional flows (copy mode)
- Up to eight-word buffers for configurable burst transfers

- Support of byte-swapping
- An available API and library of scripts
- Little-Endian and Big-Endian modes
- Hardware handshakes for low-power entry sequence
- Security support to lock contents of the SDMA script RAM.
- 4-KB ROM containing startup scripts (for example, boot code) and other common utilities that can be referenced by RAM-located scripts
- 8-KB RAM area is divided into a processor context area and a code space area used to store channel scripts that are downloaded from the system memory
- Debug support, including a OnCE port, real-time monitors, and embedded cross-trigger events
- Supported clock frequencies in process:
  - Configurable clock options for the SDMA core and the AP DMA units
    - 1:2 ratio with maximum of SDMA core running at AP Peripheral Bus speed and DMA running at max DMA frequency.
    - 1:1 ratio when both SDMA core and AP DMA clocks are set to the AP Peripheral Bus speed.
- Peripheral bus interface for configuration register programming by the AP
- The SDMA RISC engine (arithmetic and logic operations), which is referred to as the “SDMA core.”
- An internal peripheral bus connected to the Shared Peripherals Bus Interface (SPBA) that enables access to up to 14 shared peripherals. SDMA supports 32-bit accesses to word peripherals and 16-bit accesses to half-word peripherals.
- The burst DMA unit is able to perform burst accesses to the external memory
- The Burst DMA2 unit is connected to the AP Cross bar switch to Service AP Peripherals.
- All the DMA units are 32-bit AHB masters. They are connected to different buses, thus allowing concurrent accesses.

52.2 Functional Description

Figure 52-2 shows the SDMA topology, and is composed of the following components:

- SDMA Core (Section 52.3, SDMA Core.”)
- SDMA Scheduler (Section 52.4, Scheduler.”)
- Functional Units:
  - Burst DMA (Section 52.6.1, Burst DMA Unit.”)
  - Burst DMA2 Unit (Section 52.6.2, Burst DMA2 Unit.”)
- AP Control for AP control register access.
- Internal RAM and ROM Memory (Section 52.13, SDMA Programming Model)

- OnCE debug Port (Section 52.19, The OnCE Controller.")

The functional unit bus provides access by the SDMA core to the DMA units. The system bus provides access to SDMA internal memory and also supports up to 14 peripherals.

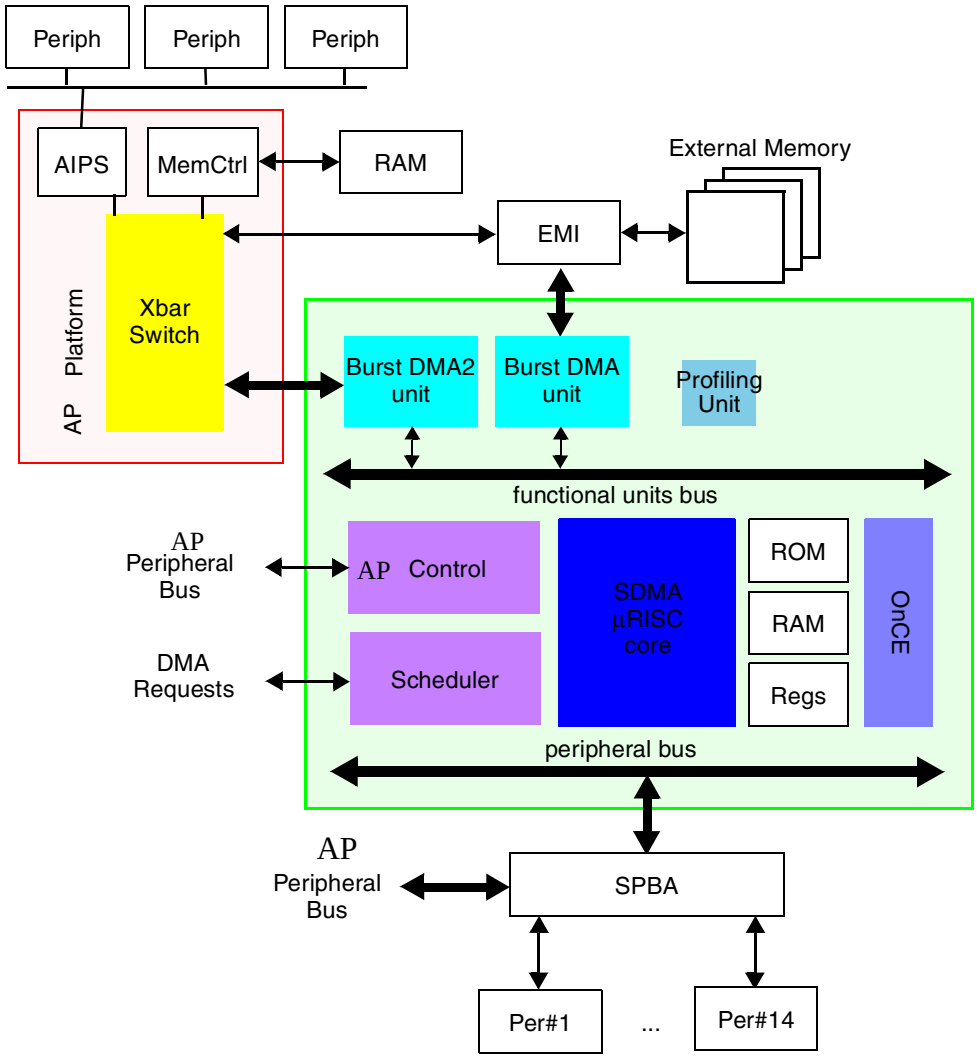


Figure 52-2. SDMA Connections

52.3 SDMA Core

The SDMA core is a customized RISC-like processor that is specifically developed to control DMA units and perform L1 tasks like byte-stuffing or framing. The SDMA core incorporates on-chip debug capability using the OnCE.

The SDMA core is based on a 32-bit register architecture with 16-bit instructions. There are eight general purpose 32-bit registers, four flags (T, LM, SF, and DF), and four PCU registers (PC, RPC, SPC, and EPC) that can address 16,384 16-bit instructions.

52.3.1 SDMA Core Structure

Figure 52-3 shows the structure of the SDMA core. It also shows the different registers, calculation resources, and possible data movements.

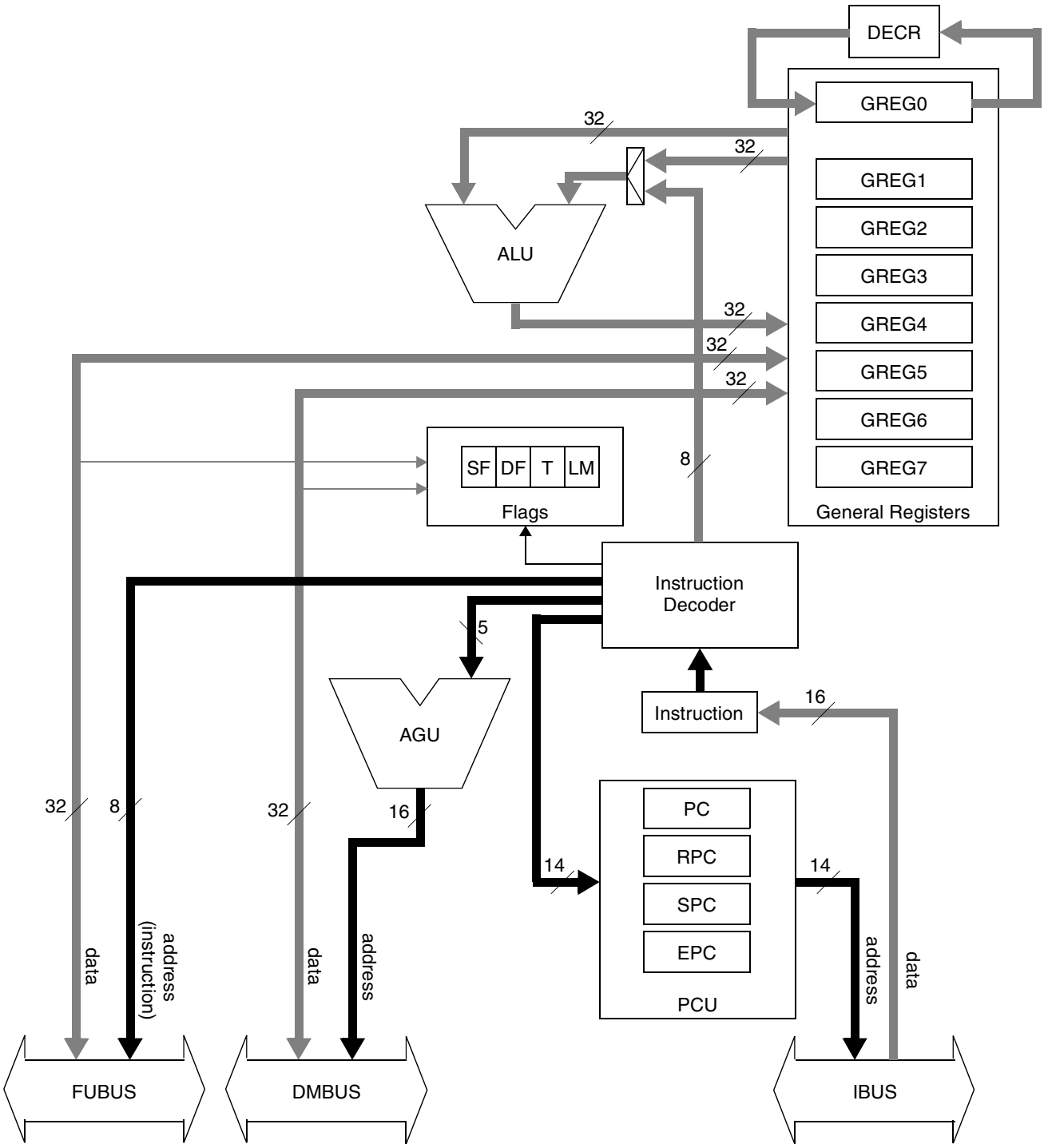


Figure 52-3. SDMA Core

- The Program Control Unit (PCU) is described in Section 52.3.2, Program Control Unit (PCU).” It handles the state of the core and generates the instruction fetch addresses. Instructions are retrieved from the Instruction Bus (IBUS) and stored in the SDMA core instruction register prior to their decoding. The PCU contains the following registers:
  - The Program Counter (PC) contains the address of the current instruction.
  - The Return Program Counter (RPC) contains the address of the instruction that follows a jump to the subroutine.
  - The Start Program Counter (SPC) contains the address of the first instruction of the current hardware loop.
  - End Program Counter (EPC) contains the address of the last instruction of the current hardware loop.
- The other core registers are the general purpose registers (GREGn) and the flags.
  - The general purpose registers can be used to hold data and addresses. They can be loaded with immediate values (for example, 8-bit data that are encoded in the instruction), results of calculations that were performed with the ALU, 32-bit data that comes from the memory or peripherals via the Data Memory Bus (DMBUS), 32-bit data that comes from the DMAs via the Functional Units Bus (FUBUS) or another general purpose register. Their content can be the operands of the ALU, the data to send on either bus (DMBUS or FUBUS), or a pointer to memory (DMBUS address).
  - The general register 0 (GREG0) is also the hardware loop counter. In hardware loops, it cannot be used for any other purpose. This register uses a dedicated decrement unit (DECR) shown in Figure 52-3.
  - The flags reflect the status of operations:
    - SF and DF are set when the last load or store on either bus (FUBUS or DMBUS) received an error response.
    - LM is set when the core is executing instructions inside a hardware loop.
    - T is set when the ALU operation result was 0 or the loop counter reaches 0 (the latter is preponderant when an ALU operation is the last instruction of a hardware loop).
- The ALU has two operands: any general register and either a second general register or an immediate value. The result is always stored into the first general register. A NOP function can be utilized by moving a register’s contents into itself (For example, the instruction: `mov R0,R0`).
- The 16-bit instructions are fetched via the instruction bus (IBUS) whose address is driven by the PC. The SDMA RAM and ROM are visible to the core as 16-bit devices through this interface.
- The memory (RAM and ROM), memory mapped registers, and external peripherals are accessed via the DMBUS. The address is always taken from a general register whose content is added to a 5-bit immediate value. This is the only available addressing mode. The DMBUS is a 32-bit data bus. Except for the peripherals that are external to the SDMA, the address accuracy is the 32-bit word (for example, adding 1 to an address points to the next word, not the next byte).
- The functional units are accessed via the FUBUS connection. The data is exchanged with any general register, but the address (which in fact is the instruction and the selector of the functional unit) comes from an 8-bit field of the corresponding load or store.

52.3.2 Program Control Unit (PCU)

This part of the SDMA core is dedicated to the control of the RISC engine as implied by the instructions that are executed. Its behavior is determined by the instruction type and the inputs of the SDMA. It contains the PC, RPC, SPC, and EPC registers that are described in Section 52.3.1, SDMA Core Structure."

52.3.2.1 Instruction Types

The state sequence and the delay of execution vary according to the type of the instruction. There are six possible categories of instructions, as follows:

1. `standard`—Most of the instructions belong to this category, and always last 1 cycle.
2. `ldf`/`stf`—These are respectively the load and store instructions that access the functional units. They last 1+n cycles where n is the number of wait-states of the targeted functional unit.
3. `ld`/`st`—These are the load and store instructions that access the memory and peripherals. They last 1+n cycles where n is the number of wait-states of the targeted device (1 for the ROM, RAM, and memory mapped registers, 1 + the external peripheral wait-states). These instructions always last at least two cycles, but the core is able to handle them in one cycle. The first wait-state is inserted outside the core.
4. `branch`—These are all the instructions that cause the Program Counter to point to another instruction other than the following one (for example, one that breaks the sequential flow). There are the absolute jumps, the conditional branches, the jump to the sub-routines, and the return from the sub-routine.
5. `loop`-modified load or store—The hardware `loop` instruction modifies the potential behavior of any load or store inside the `loop` (for example, when the LM flag is set). A jump may be implied after any such load or store if it received an error. The error causes an early exit of the loop, which means a jump to the instruction that follows the one that is pointed to by EPC. An additional cycle is required by the PCU to perform the jump (+1 to the `ld`/`st`/`ldf`/`stf` original execution delay). Although there is usually an implicit jump after the last instruction of the loop when the PC goes back to SPC, this is performed at no cycle cost.
6. `done`—The `done`, `yield`, or `yieldge` instructions are used to control channel switching. When no channel switching is performed, these instructions last a single cycle. When there is a change of channel or context switch, the delay is variable and depends on many factors (as detailed in Section 52.4.4, Context Switching").

52.3.2.2 PCU States

The PCU state is visible through outputs of the SDMA (see Section 52.20.8.4, Real-Time Debug Outputs") or the OnCE status register (see Section 52.19.4.9, OnCE Status Register (OSTAT)").

The PCU state is a four-bit field that can take the values shown in Table 52-1. Figure 52-4 shows the possible state transitions and the corresponding conditions.

Table 52-1. PCU States

Value	State	Description
0	Program	The is the usual instruction cycle.
1	Data	This state is inserted when there are wait-states during a load or a store on the data bus (ld/st type).
2	Change of Flow	This is the second cycle of any instruction that breaks the sequence of instructions (branch and done types). This state lasts only a single cycle; it is always followed by the Program state.
3	Error in Loop	This state is used when an error causes a hardware loop exit (loop-modified load or store type). This state only lasts a single cycle; it is always followed by the Program state.
4	Debug	he SDMA is stopped in debug mode.
5	Functional Unit	This state is inserted when there are wait-states during a load or a store on the functional units bus (ldf/stf type).
6	Sleep	No script is running: The core is idle after saving the last channel context.
7	Save	The context switch FSM is saving the current channel.
8	Program in Sleep	Same as Program except there is no associated channel, this state is used when instructions are executed after entering debug mode, whereas the core was in either Sleep mode.
9	Data in Sleep	This is the same as Data except there is no associated channel.
10	Change of Flow in Sleep	This is the same as Change of Flow except there is no associated channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
11	Error in Loop in Sleep	This is the same as Error in Loop except there is no associated. channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
12	Debug in Sleep	This is the same as Debug except the core was put in debug mode when no channel was active.
13	Functional Unit in Sleep	This is the same as Functional Unit except there is no associated channel.
14	Sleep after Reset	This shows that no script is running, and the core is idle after a reset. When a channel becomes active, no context is restored but the core starts its boot program located at address 0 (or the address available in register in Section 52.12.3.22, Channel 0 Boot Address (CHN0ADDR)”).
15	Restore	The context switch FSM is restoring the next channel context.

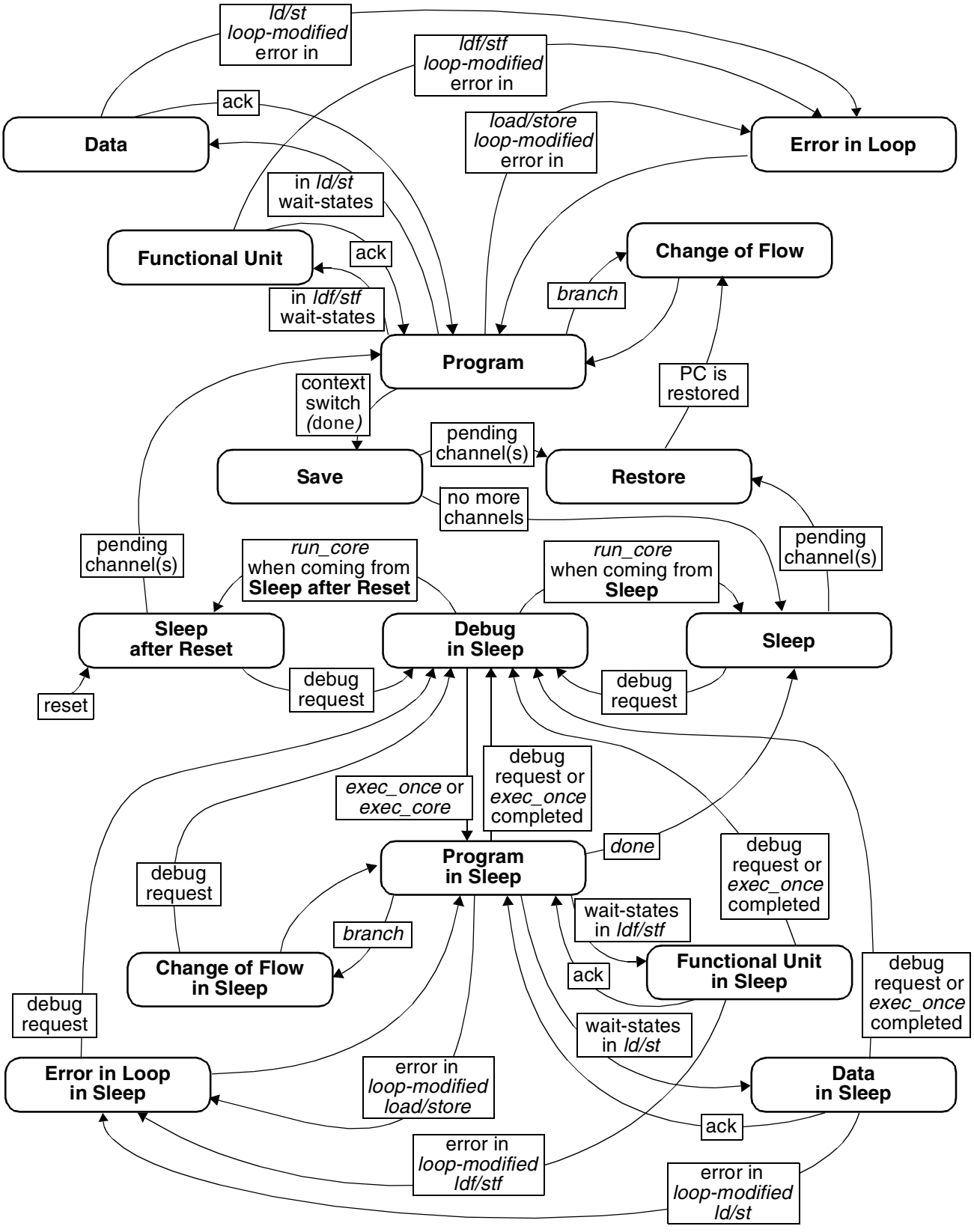


Figure 52-4. PCU State Diagram

52.3.3 SDMA Core Memory

The SDMA has two memory spaces: one for the instructions and one for the data. As both spaces share the same resources (ROM and RAM devices), the system bus manages possible conflicts when the core accesses the same resource for both an instruction read and a data read or write. Program and data memory is further described in Section 52.13.5, Address Space."

Instructions of 16-bit width are stored in 32-bit wide devices and can be accessed as data. The mapping is Big Endian: an even instruction address (terminated by 0) accesses the most significant part of the 32-bit data (bits [31:16]), and an odd instruction address (terminated by 1) accesses the least significant part of the 32-bit data (bits [15:0]). Instructions can be fetched out of internal ROM or RAM.

Data can be read from ROM, RAM, memory mapped registers, and external peripherals, and written to the same devices (except the ROM).

The ROM contains bootload scripts, channel scripts, and common subroutines which may be referenced by channel scripts elsewhere in the ROM or RAM.

The RAM is divided into a context area and a code space area which may be used to store channel scripts. The RAM contains undefined values after a hardware reset. Channel scripts and initial context values are downloaded into RAM using channel 0 which is reserved for bootload functions.

52.4 Scheduler

All channel scheduling hardware is included in the Scheduler.

52.4.1 Primary Functions

The scheduler is a hardware-based design used to coordinate the timely execution of 32 virtual DMA channels by the SDMA core on the basis of channel status and priority. The scheduler performs the following functions:

- Monitors, detects, and registers the occurrence of any one of the 48 DMA requests
- Links a specific request to a channel or group of channels (channel mapping)
- Ignores requests that are not mapped to a previously configured channel
- Maintains a list of all the channels that are requesting service
- Assigns a pre-programmed priority level (1 of 7) to every channel requesting service
- Detects and flags overrun/underrun conditions

52.4.2 Channels and DMA Requests

52.4.2.1 Channels

A Virtual Channel (hereafter simply called a channel) manages a flow of data through the SDMA. Flows are typically unidirectional. The SDMA can have up to 32 simultaneously operating channels, numbered from 0 to 31. Channel 0 is usually dedicated to control the SDMA script downloading. All the channels can be assigned by the AP software.

52.4.2.2 DMA Requests

A DMA request is caused by externally (for example, external to the SDMA) controlled conditions (for example, UART receive FIFO reaches a threshold). The SDMA currently supports up to 48 DMA requests.

52.4.2.3 Mapping from DMA Requests to Channels and Priorities

A channel can stall waiting on a single DMA request. A single DMA request can awake more than one channel (in fact, any request can awake any combination of channels). The mapping between DMA requests and channels is program-controlled. There is a storage element assigned for each of the 48 requests that contains a bitmap table of the channels that are awakened by the event. Every channel also has a three-bit register that indicates its priority.

52.4.3 Scheduler Functional Description

Section 52.4.3.1, Scheduler Overview" describes the behavior of the SDMA scheduler—from the channel enabling conditions to the highest priority pending channel selection.

52.4.3.1 Scheduler Overview

The scheduler algorithm is built in hardware. It is provided with possibilities for the AP to control its behavior. The scheduler processes incoming DMA requests, maps detected requests to 0, one, or several channels, maintains a list of channels that are requesting service (pending channels), identifies the top

priority and its associated channel, and selects the next active channel when the current channel yields. Figure 52-5 shows a functional overview.

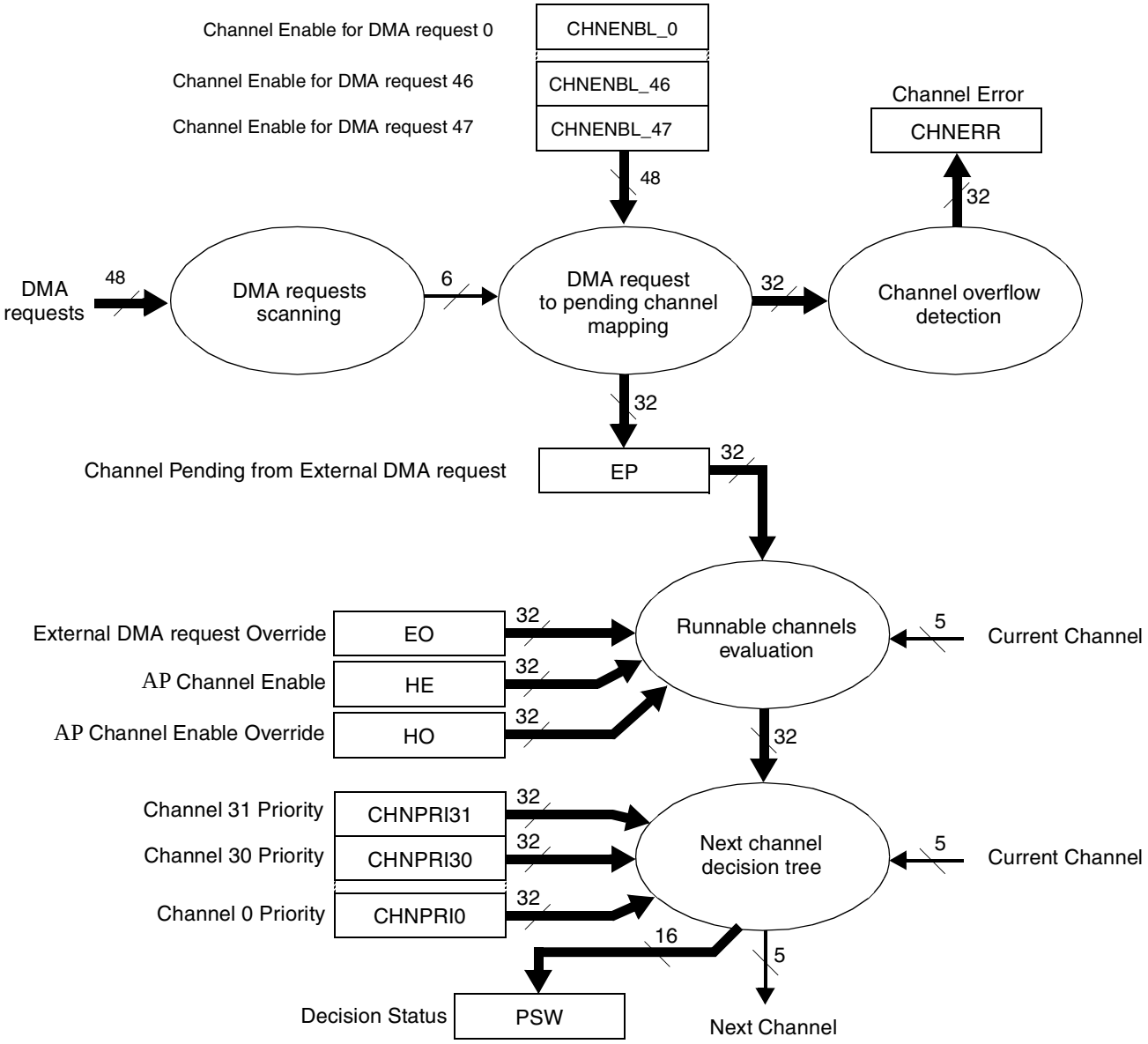


Figure 52-5. SDMA Hardware Scheduler

52.4.3.2 DMA Requests Scanning

The scheduler contains a 48-bit edge detection device that detects the rising edge of every DMA request and transmits the request number to the next stage. The DMA requests are assumed to be generated on the same reference clock as the SDMA core clock; they are detected as soon as the signal goes from a 1-to-n-cycles low state to a 1-to-m-cycles high state.

This system is able to detect single-cycle pulses as well as level-based DMA requests such as a FIFO threshold crossing. In this case, the SDMA provides a memory mapped register that can be used by the channel script to monitor the DMA requests lines, and thus determines whether the data transfer is done or not done, and then continues with the transfer or closes the channel.

When several DMA requests are detected at the same time, they are forwarded to the next scheduler stage at the rate of one request per cycle. No request is lost.

Figure 52-6 shows examples of valid DMA requests.

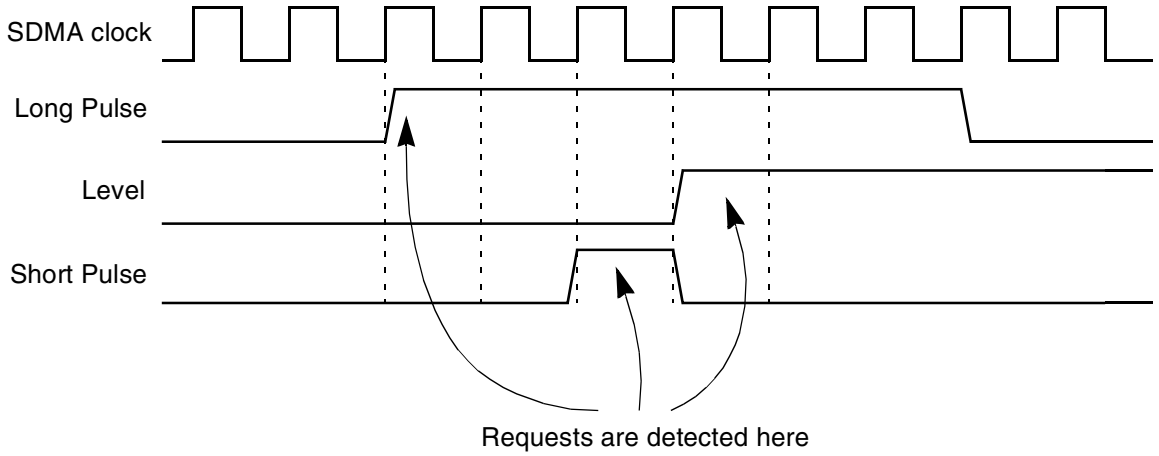


Figure 52-6. Examples of Valid DMA Requests

The DMA request inputs are connected to various sources that depend on the SoC. The exact list of DMA request inputs and their associated number is available in each respective project-specific chapter.

52.4.3.3 Mapping DMA Requests to Pending Channels

Whenever a DMA request is detected by the first stage, its number is used in the second stage to determine the channels that have to be activated. This is performed with an array of 48 registers that are 32 bits wide: There are 48 Channel Enable Registers (CHNENBLn), one register per DMA request. The DMA request number selects the Channel Enable Registers, and every bit of this 32-bit register indicates that the corresponding channel must be activated when it is a 1.

This information is passed on the EP register. For every bit of the Channel Enable Register that is set, the corresponding bit of the EP register is also set, and the remaining bits of EP are left unchanged. The transformation of EP is summarized by Equation 52-1:

$$\text{EP} = \text{EP or CHNENBLn} \qquad \textit{Eqn. 52-1}$$

The EP register is used to know which channels require service because they received a DMA request.

Typical contents of the CHNENBLn registers are all 0s, except for a single bit set. For example, a DMA request triggers one channel, but all 0s or several 1s are possible. One DMA request could activate several channels, and the channel execution sequence can be controlled by the channel priorities and numbers, as explained in the next sections. Table 52-2 illustrates an example configuration.

NOTE

From the table, the DMA request 0 is programmed to simultaneously trigger channels 0, 1, and 31. Also, DMA requests 30-47 are not used in this example. The remaining channels 2 to 30, are configured to be triggered by DMA requests 29 to 1, respectively.

Table 52-2. Channel Enable RAM Programming Example

DMA Request Number	Channel																																
	31																																0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table 52-2. Channel Enable RAM Programming Example (continued)

DMA Request Number	Channel (31 … 0)
27	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
28	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
29	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
30	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
31	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
32	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
33	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
34	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
35	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
36	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
37	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
38	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
39	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
40	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
41	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
42	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
43	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
44	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
45	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
46	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
47	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

52.4.3.4 Channel Overflow

A channel overflow occurs when a DMA request requires service from channel *n* by setting bit *n* of the register EP, but this bit is already set, meaning channel *n* is already pending. This can come from an overrun/underrun condition. This detection is possible only when the DMA requests are pulses, because a level-based DMA request stays high until it is serviced, even though an underrun or overrun condition occurs, thus preventing another edge detection of the DMA request.

The channel overflow information is saved in the 32-bit CHNERR register (1 bit per channel). You can configure the SDMA to trigger an interrupt to the AP when there are 1s in CHNERR. Every bit of CHNERR is masked with the corresponding bit of INTRMASK and if it gives a 1, the corresponding bit of INTR is set, triggering the interrupt.

52.4.3.5 Runnable Channels Evaluation

The EP register is used in conjunction with several other 32-bit registers to determine the channels that are runnable. Registers EO, DO, HO and HE, are controlled by the AP. EP is controlled by the DMA requests and their mapping to channels.

Several channels may be runnable at any given time. The i^{th} channel is runnable if (and only if) the condition below is true:

(HE[i] or HO[i]) and (DO[i]) and (EP[i] or EO[i]) ***Eqn. 52-2***

After reset, the HE[i], HO[i], EP[i], and EO[i] bits are all cleared whereas the DO[i] bits are all set. The functions associated with DO are not available for this device. When DO[i] is set, the scheduler condition becomes:

(HE[i] or HO[i]) and (EP[i] or EO[i]) ***Eqn. 52-3***

The registers in these equations are controlled as follows:

- AP (host) channel enable flag HE[i] may be set or cleared by the AP with the HSTART and STOP_STAT registers. It can also be cleared by the i^{th} channel script.

  Typical usage is for the AP to set this flag to activate the channel. The flag is cleared by the SDMA core when the transfer is done.
- Externally triggered channel pending flag EP[i] is set by the scheduler when the channel was activated by a DMA request. It can be cleared by the i^{th} channel script.
- The AP channel override flag HO[i] may be set or cleared by the AP. When set, it enables the i^{th} channel to run without the involvement of the AP.

  Typical usage is for the AP to set this flag for channels that do not need AP supervision such as channels that are controlled by DMA request events (EP).
- DO should always be set to 1 so that the runnable channel evaluation considers only HO, HE, EP, and EO.
- Externally triggered channel override flag EO[i] may be set or cleared by the AP. When set, it prevents the i^{th} channel from stopping and stalling on incoming peripheral DMA requests. This is the case when the channel is not handling data transfers with peripherals (for example, a memory to memory transfer).

The SDMA can clear the HE[i], and EP[i] bits by means of a `done` or `notify` instruction. The `done` instruction causes a reschedule; thus, enabling another channel to preempt the current one, while the `notify` instruction does not. The `done` and `notify` instructions can clear either HE[i] or EP[i] (never more than one at a time).

Table 52-3. Runnable Channel Selection Control

Register	Set by	Cleared By
HO	Write to HOSTOVR register	Write to HOSTOVR register
HE	Write to HSTART register	Write to STOP_STAT register or by the channel script with the `done` or `notify` instructions.

Table 52-3. Runnable Channel Selection Control (continued)

Register	Set by	Cleared By
DO	Write to DSPOVR register	Write to DSPOVR register
EO	Write to EVTOVER register	Write to EVTOVER register
EP	Set by external DMA request event input.	By the channel script with the `done` or `notify` instructions

52.4.3.6 Next Channel Decision Tree

The next channel number is computed from the runnable channels list, the current channel number, and their respective priorities. It is re-evaluated every cycle, but is only used when the current channel yields or terminates by executing a `yield`, `yieldge`, or `done` instruction.

The decision tree is based on the selection of the runnable channel that has the highest priority. The highest priority channel is selected according to the following rules:

- Runnable channels are sorted by priority.
- If one of the channels with the highest priority had been preempted by a channel with a higher priority, but did not want to yield to a channel of the same priority (for example, it executed a `yield`, not a `yieldge`), it is elected as the next channel.
- The channels that belong to the highest priority group are sorted by their number and the channel that has the highest number in this group becomes the next channel. For example, if priorities are the same, channel 31 will be selected before channel 30.

When the current channel requires a reschedule with a `yield`(`ge`) or a `done` instruction, the context switch decision is based on the instruction parameter, the current channel number and priority, and the next channel number and priority. The possible cases are all listed in the Table 52-4. The grayed cells in Table 52-4 correspond to unusual cases that should not occur with a typical usage of the SDMA.

Table 52-4. Channel Switching Decision with a yield, yield(ge), or done

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
`yield` (`done 0`)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Current
			Current < Next	Next[1]
	Not runnable	Not runnable	none	none[2] (occurs when the channel was disabled by the AP)
	Not runnable	Runnable	none	Next[1] (occurs when the channel was disabled by the AP)

Table 52-4. Channel Switching Decision with a yield, yield(ge), or done (continued)

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
yieldge (done 1)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Next[1]
			Current < Next	Next[1]
	Not runnable	Not runnable	none	none[2] (occurs when the channel was disabled by the AP)
	Not runnable	Runnable	none	Next[1] (occurs when the channel was disabled by the AP)
done (done >1)	Not runnable	Not runnable	none	none[2]
	Runnable	Not runnable	none	Current[3] (occurs when the done instruction does not disable the channel runnable condition)
	Not runnable	Runnable	none	Next[1]
	Runnable	Runnable	none	Current[3] (occurs when the done instruction does not disable the channel runnable condition)

1 Current channel script execution is stopped, its context is saved; the next channel context is restored and its script execution resumes

2 Current channel context is saved and SDMA enters IDLE mode

3 Current channel context is saved, then restored, and the current channel script resumes execution

Finally, when the SDMA is in IDLE mode and a runnable channel is elected as the next channel, its context is immediately restored and the script execution resumes.

The *combinatorial-decision* tree supports dynamic modifications of the EP, EO, HE, HO, and DO flags as well as dynamic modifications of the channel priorities. The propagation times are detailed in Section 52.4.3.8, Scheduler Pipeline Timing Diagram."

The decision tree status is available in the PSW register, which is continuously updated. It contains the next channel priority, the next channel number, the current channel priority, and the current channel number. When a priority is read as 0, it means the channel is not runnable.

A few examples of decisions are presented below:

- Channel 31 is running with priority 5, channels 13 and 24 are pending with the same priority 5; channel 24 is eligible as the next channel since 24 > 13.
- Channel 31 is running with priority 7, channels 13 and 24 are pending with priority 5; channel 31 is the next channel because its priority is greater than the other pending channels.
- Channels 7, 23, and 29 are pending with the same priority. Channel 7 is active and runs a yieldge; it is preempted by channel 29. After a period of time, channel 29 runs a yieldge, it is then

preempted by channel 23 that is the selected channel since channel 29 is the current channel. Later, channel 23 runs a `yieldge` and is preempted by channel 29. Channels 23 and 29 will go on switching after every `yieldge` until one of them terminates. It is only at that point that channel 7 becomes eligible again.

- Channel 11 is running with priority 3, and channel 15 is pending with priority 4. When the channel 31 script executes a `yield` instruction, it gets preempted by channel 15; then channels 6 and 18 with priority 3 become pending. Because channel 11 was preempted after executing a `yield` and there is no pending channel with a strictly greater priority, it is eligible as the next channel (although its number 11 < 18).

52.4.3.7 Scheduler State Diagram

The Figure 52-7 summarizes the behavior of the SDMA scheduler with details about the exact mechanism of the priority decision tree. It is important to understand the scheduler is a hardwired pipeline, which means all the stages are performed simultaneously every cycle, but a change on any given stage is reflected on the next stage after the delays presented in Section 52.4.3.8, Scheduler Pipeline Timing Diagram."

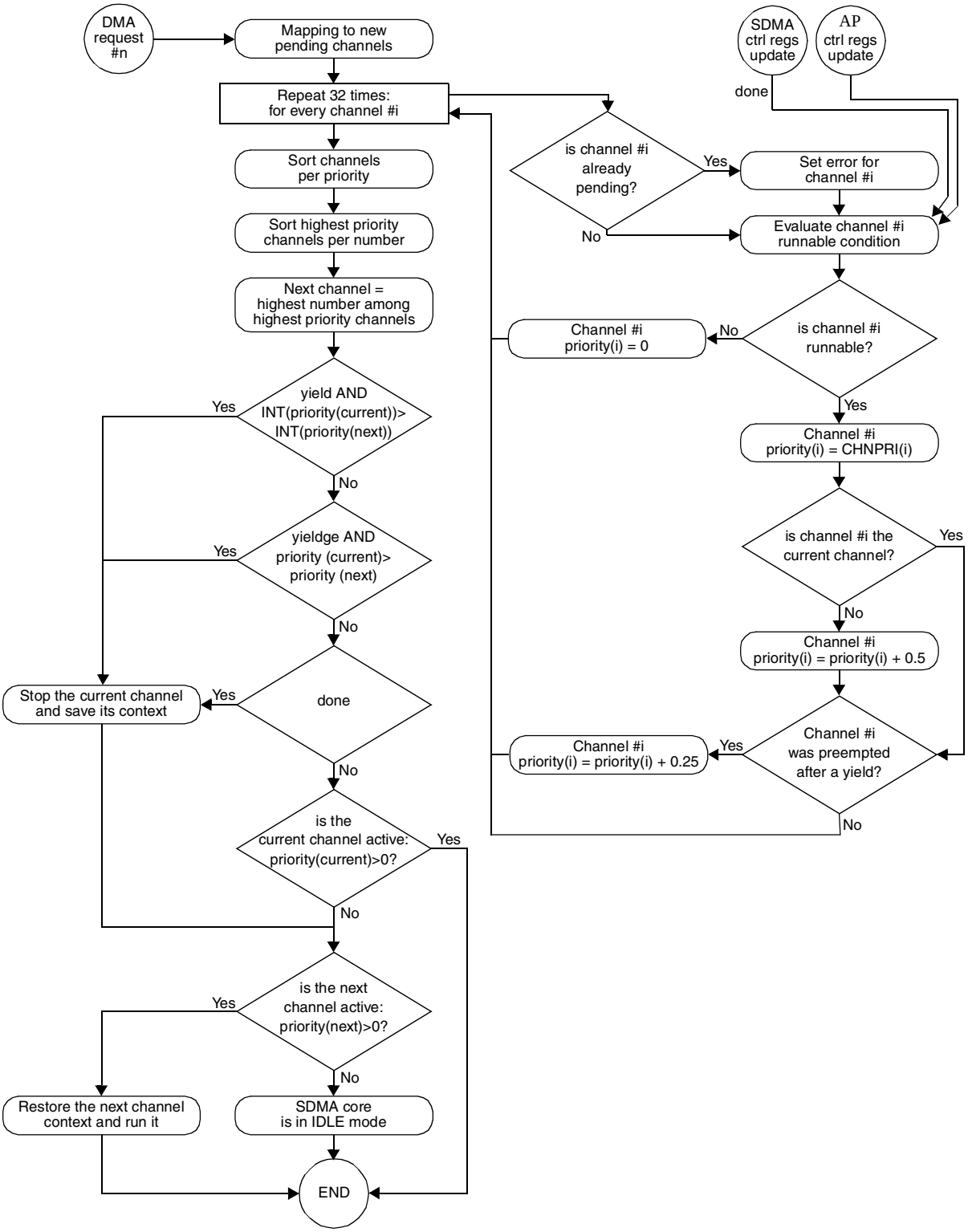


Figure 52-7. Scheduler State Diagram

52.4.3.8 Scheduler Pipeline Timing Diagram

The SDMA scheduler process of DMA-request and control-register modifications is not immediate. Figure 52-8 shows the exact delays of all the tasks. The reference clock is the SDMA core clock.

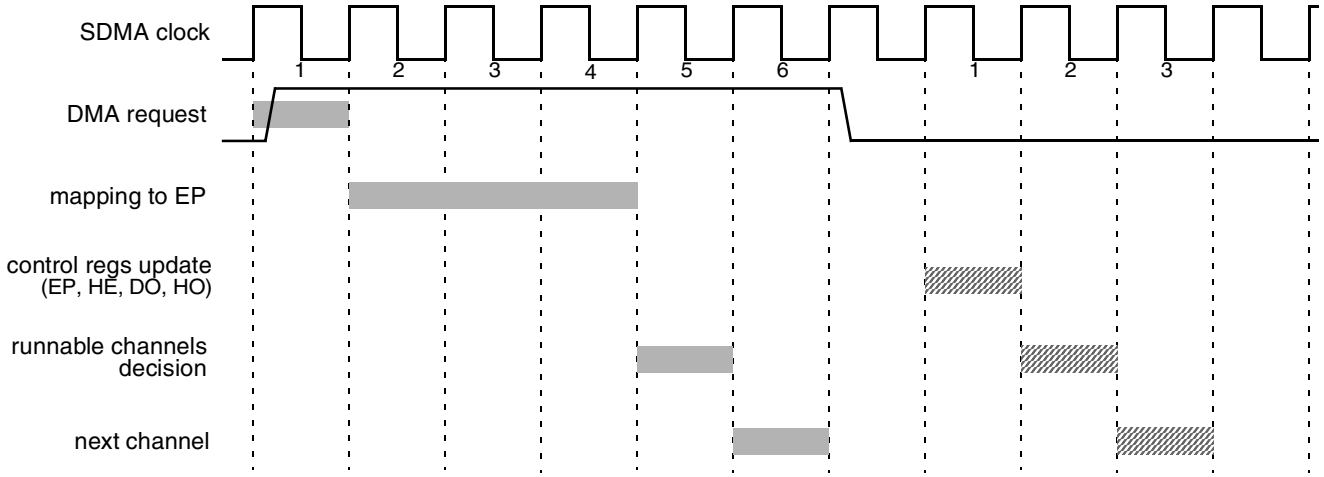


Figure 52-8. Scheduler Timing Diagram

Two numbers can be inferred from this timing diagram. First, it takes six SDMA core clock cycles to update the next channel from a DMA request. Second, it takes three SDMA core clock cycles to update the next channel from a direct modification of the condition registers (EP, DO, HE, or HO) by any processor. The processors that can modify these bits include SDMA with a `done` instruction or the AP with a write access through the corresponding control port on their respective peripheral bus).

52.4.3.9 Channel-DMA Request Mapping

The 48 DMA request inputs to the SDMA scheduler are listed in project-specific chapters. Refer to the respective chapters for this information.

52.4.3.10 Examples: How to Start a Channel

A channel can be started when Equation 52-2 is true for channel *i*:

(HE[i] or HO[i]) and (DO[i]) and (EP[i] or EO[i])

Once this equation is true, the scheduler can start this channel according to the priority of all pending channels. Several examples of configuration are listed below:

1. To start a channel triggered by AP software:
   - Initially, configure HO[i]=0, DO[i]=1, and EO[i]=1 using registers indicated in Table 52-3.
   - AP software triggers the channel by writing to the HSTART register to set HE[i]=1, thereby setting Equation 52-2 true.
2. To start a channel triggered by DMA request event.
   - Initially, configure HO[i]=1, DO[i]=1, and EO[i]=0 using registers indicated in Table 52-3.

— The DMA request is asserted to trigger the channel by setting EP[i]=1, which makes Equation 52-2 true.

52.4.4 Context Switching

On execution of a `done` or `yield(ge)` instruction, the current channel may be changed either because it has finished (which necessarily happens when the `done` instruction is executed), or it was preempted by a higher priority channel (which is possible but not systematic when the `yield(ge)` is executed).

Upon a channel change the SDMA goes through a context switch procedure. When the current channel yields or ends, the context for that channel is saved into the context RAM locations for that channel. When the next channel starts running, its context is first restored from RAM.

Since context RAM is not yet initialized by reset, there will be no context restore at the beginning of the first channel (bootload channel) run after reset. It is expected that the bootload channel will be used to initialize the context for all other channels. When the bootload channel finishes running or yields, SDMA will enter its SAVE state and save that channel's context into RAM. Then, if the bootload channel is called again later, the context will be restored from RAM when the channel starts again.

The context structure for each channel is defined in Section 52.13.4, Context Switching." and Table 52-43. There will be one context area reserved for each channel. When a channel ends or yields, the SDMA core registers are automatically saved into the context RAM and later restored from the context RAM when the channel is next run. The total RAM space reserved for 32-channel contexts is either 3K or 4K depending on whether the SMSZ bit is set in the CHN0ADDR register, which enables an additional 8 words of scratch RAM for each context.

52.4.4.1 Context Switch Modes

The exact procedure to save the context of the old channel, and to restore the context of the new channel depends on the context switch mode selected by the AP in the CONFIG control register. The following are the context switch modes:

- By default, the "dynamic" context switch is set. This mode provides the most efficient context switch for an average of eight cycles to stop the current channel, save its context, restore the next channel context, and resume its execution. It consists of saving modified registers of the current channel in the background (for example, during the channel execution)—which leaves very few registers to save when the switch is decided—resuming execution of the next channel as soon as possible (for example, when the minimal set of registers is restored), and continuing the restore phase during this execution.
- In "dynamic with no loop" mode, the same principle is followed except the modified registers are only saved in the background when the loop flag is not set. This mode offers almost the same effectiveness as the previous one, but it prevents the system from accessing the RAM during loops to save power. This is the recommended mode for an efficient context-switch when the loop bodies are short.
- In "dynamic power" mode, no background saving is performed, which reduces power consumption to the minimum. The modified registers are only saved when the context switch starts. The restore

phase is the same as before. This is the mode that achieves the optimal power consumption at the cost of a slower context-switch.

- In a “static” context switch, all the registers are saved when a context switch is decided, and all the registers are restored before starting the execution of the new channel. This mode enables a predictable behavior of the context switch since all the registers are restored prior to the channel start and all registers are saved after the channel termination.

NOTE

Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized during the context SAVE phase when the channel is done or yields. Subsequent calls to the same channel or different channels may use any of the dynamic context modes. This will ensure that all context locations for the bootload channel are initialized, and prevent undefined values in context RAM from being loaded during the context restore if the channel is re-started later.

52.4.4.2 Context Switch Procedure

The Program Control Unit goes into the *save* state, the current context is spilled into memory, and the next channel context is restored according to the context-switch mode that was selected by the AP.

The context switch procedure is as follows:

1. Load the current context’s spill base address.
2. Spill the modified registers of the current channel to memory according to the selected context switch mode while the channel is running.

   On a `done` or `yield(ge)` that causes the channel preemption, the PCU goes into the *save* state. In *static* mode, all the registers are saved; whereas, in either *dynamic* mode, the registers that were modified but not yet saved are then saved, and the PCU registers and flags are finally saved.
3. Put the SDMA core into *sleep* and wait for new channels to be serviced. This step is skipped if there are pending channels when the current channel is saved.

   As soon as there is at least one pending channel, the PCU goes into its *restore* state to restore the context of the channel that was elected by the scheduler.

   Once a channel is elected, it remains the current channel until its script requests a rescheduling operation with a `done` or `yield(ge)` instruction. That means the current channel cannot be modified by the AP, even if it is no more runnable or if its priority is modified.

   The AP can however force a reschedule by writing the corresponding bit in the CONFIG register, which has the same effect as if the script had executed a `done` instruction. That feature should only be used to stop the SDMA in emergency cases.
4. Load the context base-address of the new channel.

   In “static” mode, all the registers are restored. In either “dynamic” modes, only the PCU registers are restored.

   The new channel is running. In “static” mode, no more activity regarding context restoring or saving is performed. In either “dynamic” modes, the registers are restored in the background every time an access to the context RAM is possible, and priority is given to restoring the registers that

are required by the next instruction to be executed. When a register has not been restored and the next instruction needs it, this instruction gets stalled until the register was restored.

In “dynamic” and “dynamic with no loop” modes, background saving of dirty registers is performed every time an access to the context RAM is possible and allowed by the context switch mode.

NOTE

The contents of a channel context space in the context RAM depends on the selected context switch mode. In “dynamic” and “dynamic with no loop” modes, the contents of the context RAM tend to match the contents of the SDMA registers (except for the PCU registers and flags that are never saved in the background). In “dynamic power” and “static” modes, the contents of the context RAM remain unchanged until the channel terminates with a done or gets preempted.

52.4.4.3 Context Map in Memory

Refer to Section 52.13.4, Context Switching.”

52.5 Profiling Unit

Profiling Unit in SDMA is used for measuring certain performance related numbers which can be used to do performance analysis of SDMA. This will be useful for doing S/W based profiling of SDMA. This Unit contains 5 separate counters. All the counters are running at SDMA Core Clock. All these counters can be programmed individually to measure one of the following values:

- Number of cycles when SDMA is in Sleep State.
- Number of cycles when one of the DMA is in Sleep State.
- Number of cycles for which a particular channel is active.
- Number of cycles for which a particular channel is runnable but it is pending due to some higher priority channels.
- Number of cycles for which SDMA core is in FUBUS state. For understanding FUBUS state refer PCU States.
- Number of cycles for which one of the DMA request is asserted but it is pending to be serviced.
- Number of cycles for which a particular DMA is in sleep state while running a particular channel

One additional counter (6th) is there which has an extra feature in addition to the above mentioned ones. It can be used as a free running counter which can be enabled both by SDMA core as well as AP core. All the counters are 32 bit registers whose 9 bits (31:23) are used to program the counter, 23rd bit is the over flow bit and the remaining 22 bits (21:0) are used as counter. For details of the programming refer to Profile Counter Registers (PRF_CNT_x)

Another register is Profile Status/Config register. This register contains “Profile Enable” (PRF_CFG[0]) bit and “Interrupt enable” (PRF_CGF[6:1]) bits for all the counters. For details refer to Profile Config/Status Register (PRF_CGF)

52.5.1 Profiling Sequence

The profiling sequence is as follows:

1. Program the 9 bit of the Registers (PRF_CNT_x[31:23]) corresponding to the counters that need to be enabled.
2. To enable the Interrupt to be sent to AP core, program the Interrupt Enable bits for the corresponding counters.
3. Run the software that needs to be profiled and enable the Profile Enable bit when the profiling has to be started. All the counters, OFL, and ISR bits of Profile Config/Status Register (PRF_CGF) will be reset as soon as the Profile Enable bit is set (rising edge).
4. The interrupt will be sent to AP core if any of the counter overflows and its Interrupt Enable bit is set in the Profile Config/Status Register (PRF_CGF).
5. Interrupt service Routine in the AP core has to disable the Profile Enable bit in order to stop all the counters. Counters keep on running even after sending the interrupt to AP core.
6. Interrupt Service Routine in AP core has to clear the ISR bit of Profile Config/Status Register (PRF_CGF). This bit is W1C bit.
7. After completing steps 5 and 6, the AP core can read any of the 6 registers to record the profiling values for the software.

NOTEs

While running the sixth Register in Free Running Counter mode, SDMA core can set/reset the EN bit of Profile Config/Status Register (PRF_CGF). This EN bit should be set/reset after the Profile Enable bit is set (in step 3). Setting/Resetting this bit will start/stop the counter respectively.

Disabling of all the counters can be done at any point of time while the profile software is running. When the profile enable bit is cleared, all counters stop and its value freezes. All the counters rerun from the same value again if the Profile Enable bit is set.

52.6 Functional Units

The functional units are small systems that are used by the SDMA core to handle data transfers between the core and a bus domain external to the SDMA.

The SDMA core is able to control and exchange data with these systems by sending instructions and reading or writing data from/to the functional units' registers via the FUBUS. This is done with the `ldf` and `stf` instructions.

The following sections provide introductions to the available functional units. Section 52.18, Functional Units Programming Model" provides descriptions the functional units' behaviors.

52.6.1 Burst DMA Unit

The burst DMA unit enables the SDMA core to perform data transfers to and from the AP memory. It is optimized for accessing SDRAM-like devices. It does not provide control to assign a privilege level to the DMA access. The burst DMA unit provides the SDMA with means to do the following:

- Perform up to 8-beat read and write bursts to the AP memory, which optimizes throughput when accessing SDRAM-type devices because of an internal, 36-byte FIFO
- Access the AP memory at once or twice the SDMA core frequency
- Copy data from one AP memory location to another AP memory location at the AP bus speed, which provides a very high throughput
- Control the method for addressing the AP memory (automatic increment of addresses or frozen addresses—the former aimed at accessing RAM-like memory and the latter aimed at accessing single-address FIFOs)
- Enable or disable automatic prefetch when reading data from the AP memory. When the prefetch mode is selected, the burst DMA automatically triggers external bursts to fill its FIFO without waiting for the SDMA core to request the corresponding data, greatly improving throughput.
- Rely on the DMA to automatically flush its FIFO content when there is enough data to generate an 8-beat burst to the AP memory. Or, it forces a flush when a data transfer must terminate.

  In the former case, the SDMA core may only be stalled when it tries writing data and there is not enough room left in the FIFO. In the latter case, the core is stalled until the data is effectively written to the AP memory.

  In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the AP memory. This error status is retrieved by a later access to the burst DMA. Terminating a write data transfer with a forced flush command guarantees that any bus error to the AP memory is caught.
- Handle address alignment issues between the AP memory map and the SDMA core data. This enables the core to read or write 32-bit data from the burst DMA, whereas the corresponding AP address is not 32-bit aligned. This drastically improves the SDMA scripts' efficiency since the same loop that transfers 32 bits at a time can be used regardless of the start and end addresses in the AP memory space.

This unit structure and registers are described in Section 52.6.1.1, Burst DMA Structure" and Section 52.6.1.2, Burst DMA Registers."

52.6.1.1 Burst DMA Structure

The burst DMA is depicted in Figure 52-9. It is essentially made up of a 36-byte FIFO, address registers, and a controlling state-machine. The 36-byte FIFO enables eight-word buffering with address alignment, and the state-machine manages clock adaptation when required.

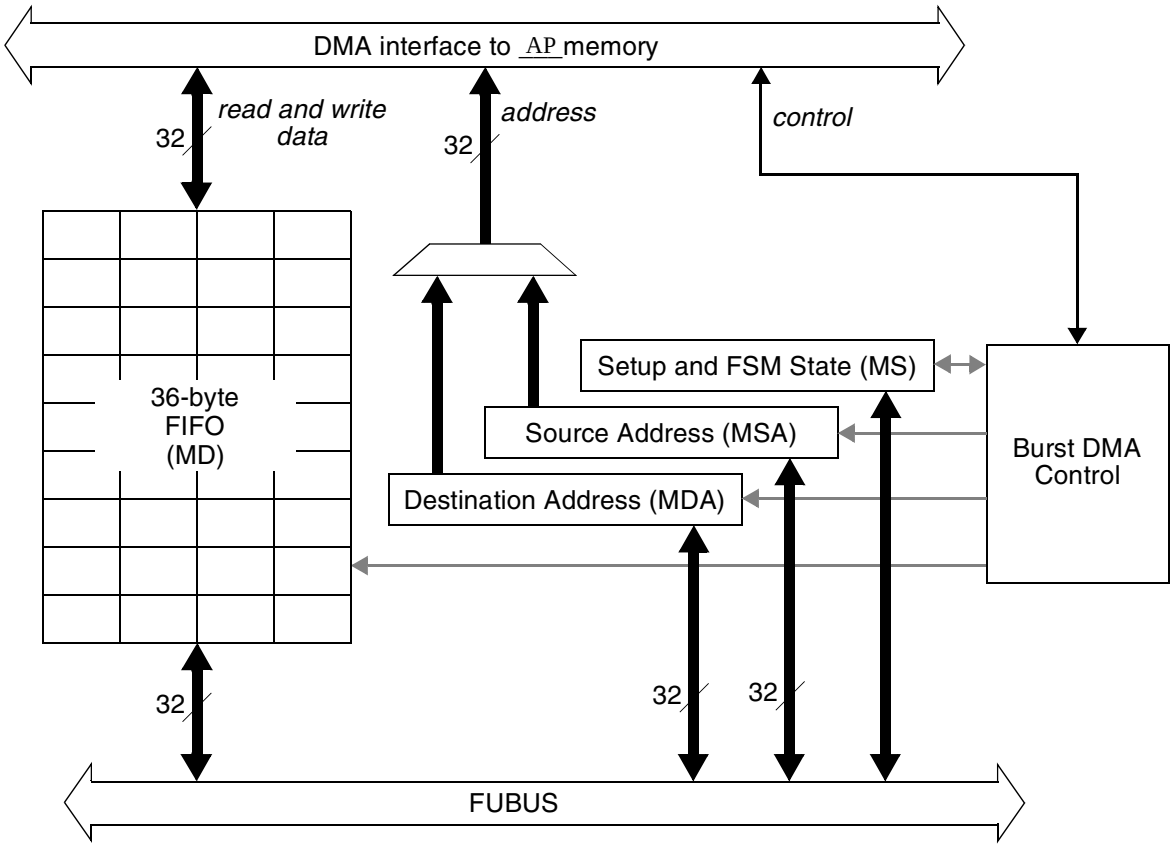


Figure 52-9. Burst DMA Structure

52.6.1.2 Burst DMA Registers

There are four registers, as follows, that may be accessed from the SDMA core:

- MSA (Memory Source Address) — Holds the source byte address in the AP memory map for reading data from this location. This register is automatically modified every time the core reads new data from the FIFO.
- MDA (Memory Destination Address) — Holds the destination byte address in the AP memory map for writing data to this location. This register is automatically modified every time the core writes new data into the FIFO.
- MD (Memory Data) — Labels the 36-byte FIFO access point: Reading a byte, halfword, or word from MD respectively retrieves the first 1, 2, or 4 bytes of the FIFO (for example, the bytes that were stored first by the DMA state-machine when transferring data from the AP memory).

  When the FIFO does not hold as many bytes as required by the SDMA core, the core is stalled until the missing bytes are read from the AP memory. In the case of prefetch mode, the DMA controller decides when it should start a burst to AP memory in order to reduce the risk to not have the required data for the future accesses of the core. When there is no prefetching, a burst is triggered when the required data is not available in the FIFO.

  Writing a byte, halfword, or word to MD stores 1, 2, or 4 bytes, respectively, at the end of the FIFO (for example, these bytes are transmitted to the AP memory after all the other bytes that were

previously stored in the FIFO). When the FIFO does not have enough room left to hold the written data, the SDMA core is stalled until a sufficient amount of FIFO contents are flushed out to the AP memory. Flushing is decided by the DMA controller when there are enough bytes in the FIFO to perform the largest allowed burst to AP memory (the exact size depends on the burst start address and the AHB 1 Kbyte boundary rule). However, the SDMA core has the ability to force the flushing operation at any time, for example, when at the end of the data transfer, prior to channel closure.

- MS (Memory Setup) — Contains the state of the burst DMA control, the two flags that define whether each address register is incremented after every access to the external memory, and another flag that is set when a bus error occurred.

52.6.1.3 Data Transfers

Three typical usages have been identified that involve the burst DMA: the data transfer startpoint, the endpoint, or both. Every case requires a different procedure, as listed in the following sections:

52.6.1.3.1 Data Retrieval from the AP Memory

The following steps retrieve data from AP memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the source address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the source address register itself (MSA).
- Read data from the FIFO using the *ldf MD* instruction as many times as needed. If an error occurred during the fetch from AP memory, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from the FIFO.

52.6.1.3.2 Storing Data Into the AP Memory

The following steps store data from AP memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the destination address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the destination address register itself (MDA).
- Store data into the FIFO using the *stf MD* instruction as many times as needed.
- When the transfer is finished and if the DMA worked in automatic flush mode, force the flush of the FIFO. This instruction is stalled until all the FIFO data is effectively sent to the AP memory and the error status of the transfer is available in the DF flag.

52.6.1.3.3 Transferring Data Between Two AP Memory Locations

The following steps copy data between two AP memory locations using the burst DMA unit:

- Set up the MS flags to reflect the modes for the source and destination addresses (all the combinations are possible), then initialize the source address register (MSA) and the destination address register (MDA). Both addresses must be word-aligned.
- Use as many *stf MD* instructions with the *COPY* flag as needed. Every instruction triggers a burst read of a given number of words from the source address (this number is provided to the burst

DMA via the SDMA core general purpose register, which is referenced in the *stf* instruction). Once all the data is loaded into the FIFO, the DMA empties it with a write burst of the same count to the destination address. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.

- Once the transfer is done, there should be a final access to the burst DMA to check the error status.

52.6.2 Burst DMA2 Unit

The second burst DMA2 unit is the replica of the first Burst DMA unit. This DMA is used to connect BP side memory. It is optimized for accessing SDRAM-like devices. It does not provide control to assign a privilege level to the DMA access. The burst DMA2 unit provides the SDMA with means to do the following:

- Perform up to 8-beat read and write bursts to the BP memory, which optimizes throughput when accessing SDRAM-type devices because of an internal, 36-byte FIFO
- Access the memory at once or twice the SDMA core frequency
- Copy data from one BP memory location to another BP memory location at the BP bus speed, which provides a very high throughput
- Control the method for addressing the BP memory (automatic increment of addresses or frozen addresses—the former aimed at accessing RAM-like memory and the latter aimed at accessing single-address FIFOs)
- Enable or disable automatic prefetch when reading data from the BP memory. When the prefetch mode is selected, the burst DMA2 automatically triggers external bursts to fill its FIFO without waiting for the SDMA core to request the corresponding data, greatly improving throughput.
- Rely on the DMA to automatically flush its FIFO content when there is enough data to generate an 8-beat burst to the BP memory. Or, it forces a flush when a data transfer must terminate.

  In the former case, the SDMA core may only be stalled when it tries writing data and there is not enough room left in the FIFO. In the latter case, the core is stalled until the data is effectively written to the memory.

  In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the memory. This error status is retrieved by a later access to the burst DMA2. Terminating a write data transfer with a forced flush command guarantees that any bus error to the memory is caught.
- Handle address alignment issues between the memory map and the SDMA core data. This enables the core to read or write 32-bit data from the burst DMA2, whereas the corresponding BP address is not 32-bit aligned. This drastically improves the SDMA scripts' efficiency since the same loop that transfers 32 bits at a time can be used regardless of the start and end addresses in the BP memory space.

This unit structure and registers are described in Section 52.6.2.1, Burst DMA2 Structure" and Section 52.6.2.2, Burst DMA2 Registers."

52.6.2.1 Burst DMA2 Structure

The burst DMA2 is depicted in Figure 52-10. It is essentially made up of a 36-byte FIFO, address registers, and a controlling state-machine. The 36-byte FIFO enables eight-word buffering with address alignment, and the state-machine manages clock adaptation when required.

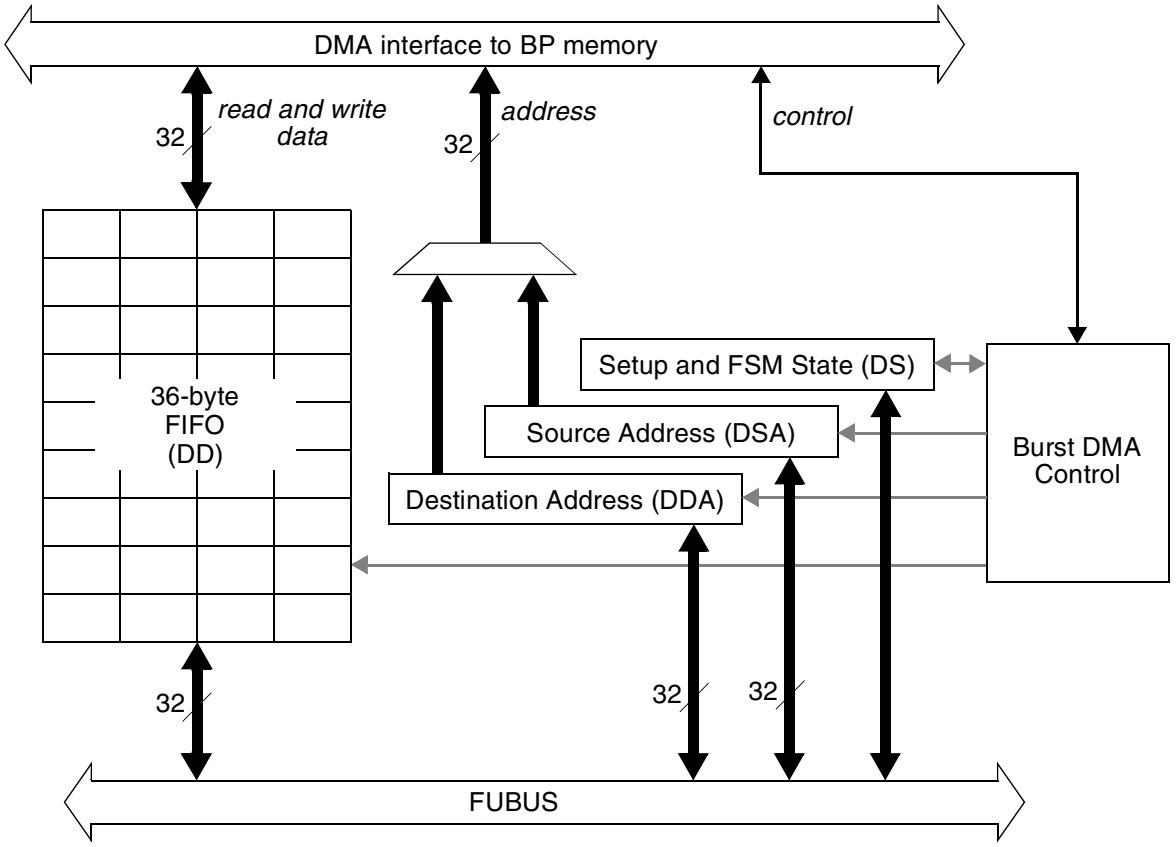


Figure 52-10. Burst DMA2 Structure

52.6.2.2 Burst DMA2 Registers

There are four registers, as follows, that may be accessed from the SDMA core:

- DSA (Memory Source Address) — Holds the source byte address in the memory map for reading data from this location. This register is automatically modified every time the core reads new data from the FIFO.
- DDA (Memory Destination Address) — Holds the destination byte address in the memory map for writing data to this location. This register is automatically modified every time the core writes new data into the FIFO.
- DD (Memory Data) — Labels the 36-byte FIFO access point: Reading a byte, halfword, or word from DD respectively retrieves the first 1, 2, or 4 bytes of the FIFO (for example, the bytes that were stored first by the DMA state-machine when transferring data from the memory).

  When the FIFO does not hold as many bytes as required by the SDMA core, the core is stalled until the missing bytes are read from the BP memory. In the case of prefetch mode, the DMA controller

decides when it should start a burst to BP memory in order to reduce the risk to not have the required data for the future accesses of the core. When there is no prefetching, a burst is triggered when the required data is not available in the FIFO.

Writing a byte, halfword, or word to DD stores 1, 2, or 4 bytes, respectively, at the end of the FIFO (for example, these bytes are transmitted to the BP memory after all the other bytes that were previously stored in the FIFO). When the FIFO does not have enough room left to hold the written data, the SDMA core is stalled until a sufficient amount of FIFO contents are flushed out to the memory. Flushing is decided by the DMA controller when there are enough bytes in the FIFO to perform the largest allowed burst to memory (the exact size depends on the burst start address and the AHB 1 Kbyte boundary rule). However, the SDMA core has the ability to force the flushing operation at any time, for example, when at the end of the data transfer, prior to channel closure.

- DS (Memory Setup) — Contains the state of the burst DMA2 control, the two flags that define whether each address register is incremented after every access to the external memory, and another flag that is set when a bus error occurred.

52.6.2.3 Data Transfers

Three typical usages have been identified that involve the burst DMA2: the data transfer startpoint, the endpoint, or both. Every case requires a different procedure, as listed in the following sections:

52.6.2.3.1 Data Retrieval from the BP Memory

The following steps retrieve data from BP memory using the burst DMA2 unit:

- Set up the DS flags to reflect the mode for the source address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the source address register itself (DSA).
- Read data from the FIFO using the *ldf DD* instruction as many times as needed. If an error occurred during the fetch from BP memory, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from the FIFO.

52.6.2.3.2 Storing Data Into the BP Memory

The following steps store data from BP memory using the burst DMA2 unit:

- Set up the DS flags to reflect the mode for the destination address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the destination address register itself (DDA).
- Store data into the FIFO using the *stf DD* instruction as many times as needed.
- When the transfer is finished and if the DMA worked in automatic flush mode, force the flush of the FIFO. This instruction is stalled until all the FIFO data is effectively sent to the BP memory and the error status of the transfer is available in the DF flag.

52.6.2.3.3 Transferring Data Between Two BP Memory Locations

The following steps copy data between two BP memory locations using the burst DMA2 unit:

- Set up the DS flags to reflect the modes for the source and destination addresses (all the combinations are possible), then initialize the source address register (DSA) and the destination address register (DDA). Both addresses must be word-aligned.
- Use as many *stf DD* instructions with the *COPY* flag as needed. Every instruction triggers a burst read of a given number of words from the source address (this number is provided to the burst DMA2 via the SDMA core general purpose register, which is referenced in the *stf* instruction). Once all the data is loaded into the FIFO, the DMA empties it with a write burst of the same count to the destination address. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.

Once the transfer is done, there should be a final access to the burst DMA2 to check the error status

52.7 SDMA Security Support

The SDMA provides support to SDMA software to block unauthorized updates to the scripts in RAM.

SDMA supports the following Security modes:

- Open Mode: has full control to load scripts and context into SDMA RAM. This is the default mode.
- Locked Mode: The AP loads scripts and channel contexts at startup when it is still executing known safe software. When finished, it locks the SDMA to prevent further updates to RAM and selected registers. More details described in Section 52.7.1, Locked Mode."

52.7.1 Locked Mode

The LOCK bit in the SDMA_LOCK register provides support for SDMA scripts to freeze RAM contents after the initial bootload routine to prevent future unauthorized updates to SDMA RAM. After initial RAM contents are uploaded, AP software can set the LOCK bit to secure the RAM contents to prevent future updates by an unauthorized. After the LOCK bit is written with a '1', the SDMA is "locked" until reset.

The LOCK bit can be read in the SDMA's internal memory map in the LOCK register (see Section 52.14.3.20/52-106). SDMA scripts which load information into RAM can check the value of the LOCK bit to determine if an upload to RAM is allowed. If not allowed, the script can refuse to allow the request to copy data into the RAM to continue. The exact use of the LOCK bit in SDMA scripts for security control will be described in SDMA software documentation (see Section 52.25, References").

While SDMA is locked, attempts to write to the SDMA_LOCK, CHN0ADR, ILLINSTADDR, and ONCE_ENB registers will be ignored. All registers remain readable. Writes to other registers are still allowed.

Once the SDMA is locked, the LOCK bit can only be cleared by a reset. A hardware reset will always clear the LOCK bit. A software reset initiated by writing to the RESET register will only clear the LOCK bit if the SRESET_LOCK_CLR bit in the SDMA_LOCK register is set. Since SDMA_LOCK register cannot be updated if SDMA is locked, the SRESET_LOCK_CLR bit must be configured before setting the LOCK bit. The SREST_LOCK_CLR bit will also be cleared by resets that clear the LOCK bit.

The SDMA RISC core uses the ILLINST and CHN0ADDR registers as pointers to determine where to jump to after an illegal instruction or upon boot after a reset. The LOCK bit prevents updates to these registers to protect against unauthorized changes to these pointers.

While SDMA is locked, the ONCE_ENB register cannot be written to prevent the OnCE under AP control from being used to gain access to SDMA internal memory. If AP control of the OnCE is enabled before setting the LOCK bit, the AP can use the ONCE for debug purpose after LOCK is set.

52.8 OnCE and PCU Debug States

The SDMA has two different debug modes in which the OnCE performs debug instructions. Refer to Figure 52-4 for an example of the PCU states in debug. The following are the two debug states:

- When a channel is running (that is, when CCR and CCPRI are different from 0, which can be read in the PSW register), SDMA can execute a `SoftBkpt` instruction from the channel script or receive a debug request. When either happens, the SDMA enters its "Classical" *Debug* state, which is described in Section 52.18.3, OnCE and Real-Time Debug."
- When a channel is not running, the SDMA can be in *Sleep* state or in *Sleep after Reset* state. If a debug request is sent to the core, it enters its *Debug in Sleep* state. This debug mode works similarly to the "Classical" *Debug* state, except it returns to the original state (*Sleep* or *Sleep after Reset*) when the debug mode is left via the `exec_core` instruction of the OnCE. From this *Debug in Sleep* state, the SDMA can execute a program whereas no channel is running. If a new debug request is sent to the core or if a `SoftBkpt` is executed, it comes back to this *Debug in Sleep* state.

The OnCE is provided with several instructions that can be executed when the core is in either debug state. Table 52-5 summarizes the behavior of these OnCE debug instructions. There exists other secondary OnCE instructions that are described in Section 52.18.3, OnCE and Real-Time Debug."

Table 52-5. SDMA in Debug Mode

Instruction	Debug	Debug in Sleep
`exec_once`	`exec_once` <instruction> SDMA executes the <instruction> and returns to the *Debug* state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.	`exec_once` <instruction> SDMA executes the <instruction> and returns to the *Debug in Sleep* state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.
`run_core`	`run_core` <instruction> SDMA executes the <instruction>, leaves the *Debug* state and continues executing the channel script from the position where it stopped. This command must not be used with an instruction that modifies the PC value.	`run_core` <instruction> SDMA executes the <instruction> and returns to its *Sleep* or *Sleep after Reset* initial state. This command must not be used with an instruction that modifies the PC value.
`exec_core`	`exec_core` <instruction> It is similar to `run_core` except it requires an instruction that changes the PC value (jump, branch...): the SDMA jumps to the new PC value, leaves the *Debug* state and starts executing instructions from this new PC value.	`exec_core` <instruction> If the previous state was *Sleep after Reset*, the SDMA returns to this state, and Chn0Addr value overrides the PC value. Otherwise, the SDMA jumps to the new PC value and starts executing instructions from this new PC.

NOTE

The feature `exec_core` in *Debug in Sleep* after *Sleep after Reset* was added for the Channel boot (channel 0) to allow the debugger to return to *Sleep after Reset* state with a new PC value. The SDMA will be ready to boot at the Chn0Addr address.

52.9 SDMA Clocks and Low Power Modes

The SDMA receives several root clocks from the SoC clock controller module and performs adaptive clock gating to optimize its power consumption. From a user standpoint, clock gating and power mode selection are fully automatized inside the SDMA. Root clock control is available from the SoC clock controller module.

There are numerous clock sources that are used in the SDMA. They belong to one of two possible clock domains listed in Table 52-6, and have frequency constraints within each domain. Clocks are considered asynchronous between domains.

Within the AP/SDMA clock domain, all clocks must come from the same PLL. The AP DMA interfaces (burst DMA) receive their clock from the AP DMA clock source whose frequency can be once or twice the frequency of the SDMA core clock. The DMA interfaces are designed to work at the AP DMA frequency, but the SDMA core is physically limited to a maximum 104 MHz frequency. Since this is lower than the maximum AP DMA frequency, the SDMA core clock is tied to the AP peripheral clock frequency.

The AP Peripheral Bus Clock source must be an exact sub-frequency of the SDMA Core clock source (any integer value greater or equal to 1).

Table 52-6. Clocking Scheme

Clock Domain	Source Clock	Comments
AP	SDMA core (SDMA main core)	Source clock for the core and all its operations; this clock is thus used by most of the SDMA sub-modules.
	AP DMA	DMA interface for the burst DMA. It is balanced with the main clock source, and its frequency is either once or twice the main clock frequency.
	AP peripheral	Connection to the AP peripheral bus. It is a sub-frequency of the main clock frequency.
Burst DMA2	AP DMA	DMA interface for the Burst DMA2. It is balanced with the main clock source, and its frequency is either once or twice the main clock frequency.
	BP peripheral	Connection to the BP peripheral bus; no constraint regarding its frequency but it is usually a sub-frequency of the BP DMA clock frequency.
JTAG	TCK	Clock for JTAG access, limited to maximum of 1/8 of the SDMA core clock frequency.

The JTAG clock is sampled by the SDMA main clock to determine its rising edge. This simplifies design and clock management, but it also adds a ratio constraint between those two clocks. It is guaranteed the

JTAG interface works properly when the frequency of TCK is lower than 1/8th of the frequency of the SDMA main clock (which is about 8 MHz when the SDMA core clock frequency is 66 MHz).

52.9.1 Clock Gating and Low Power Modes

The SDMA automatically performs power saving without requiring user involvement. It implements two levels of automatic clock gating.

52.9.1.1 Coarse Clock Gating

Every sub-module clock comes from one of the five available sources, and is gated with the sub-module specific enabling condition. Table 52-7 displays the sub-module clocks and their source. It also indicates the relationships that may exist between different sub-modules clock enables.

Table 52-7. Sub-Modules Clocks

Sub-Module	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
Core	SDMA Main Core	The core sub-module clock is running when the core is not in one of its sleep states (Sleep or Sleep after Reset) or there is a pending channel. Typically, the core sub-module clock is stopped once all the channels are processed and the core enters its sleep state. A new pending channel awakes the core sub-module clock.	None
Memories	SDMA Main Core	The clock activation only occurs during a core access.	Disabled when Core sub-module clock is disabled or no memory access in progress
Scheduler	SDMA Main Core	Its clock only runs when scheduling is needed: for example, when there are pending channels, upon reception of a DMA request, and anytime the AP modifies the channel running conditions.	None
AP Control	SDMA Main Core & AP peripheral	The AP peripheral clock is solely used to determine the frequency ratio with the SDMA main clock. The control registers' clock is based on SDMA main clock; it is active when the AP or the SDMA modifies the contents of one of these registers.	None
Burst DMA	SDMA Main Core & AP DMA	The burst DMA has two clocks: The first clock is derived from the SDMA main core clock and drives registers that are connected to the FUBUS. The second clock is derived from the AP DMA clock and drives registers that are connected to the AP DMA bus outside the SDMA. Both clocks are enabled during active phases of data transfers (for example, these clocks are turned off when the burst DMA is not used by the running channel script).	Disabled when Core sub-module clock is disabled

Table 52-7. Sub-Modules Clocks (continued)

Sub-Module	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
Burst DMA2	SDMA Main Core & AP DMA	The burst DMA2 has two clocks: The first clock is derived from the SDMA main core clock and drives registers that are connected to the FUBUS. The second clock is derived from the AP DMA clock and drives registers that are connected to the AP DMA bus outside the SDMA. Both clocks are enabled during active phases of data transfers (for example, these clocks are turned off when the burst DMA2 is not used by the running channel script).	Disabled when Core clock is disabled
OnCE	SDMA Main Core	The OnCE clock is derived from main source clock. It is disabled by default. In order to use the OnCE, its clock must be explicitly turned on, either by enabling the OnCE access from the AP peripheral bus (register ONCE_ENB), or by driving the clk_gating_off input pin high. This is a SDMA input whose driver depends on the SoC implementation (typically a JTAG controller). The OnCE also receives the TCK input, which is the JTAG clock. It does not use it as a functional clock; the TCK input is sampled instead. Refer to Section 52.19.5.2, Synchronization Implementation."	When enabled, all other clocks are systematically on (clock gating is off)

52.9.1.2 Refined Clock Gating

The SDMA implements a second level of clock gating on a register-per-register basis. Unlike the first level that covers all the SDMA flip-flops, except the synchronizers (only five flip-flops are always running), the second level is only available for eligible registers, which amounts to about 90% of the SDMA flip-flops.

These gated registers are only clocked when the hardware logic detects a new data loading. This additional gating further reduces dynamic power consumption.

52.9.1.3 Low Power Modes and User Control

Power savings are automatically managed by the SDMA hardware without any user involvement; however, one can distinguish three different power modes: SLEEP, RUN, and DEBUG. The following Table describes these modes, and shows how to switch from one mode to another.

Table 52-8. Power Modes

Power Mode	Sub-modules							Comments
	Core	Memories	Scheduler	AP Control	Burst DMA	Burst DMA2	OnCE	
SLEEP	off[1]	off	wait[2]	wait	off	off	off	Set when the PCU state is either *Sleep* or *Sleep after Reset* and the SDMA is not in DEBUG mode. This is the default mode after reset.
RUN	on[3]	wait	wait	wait	wait	wait	off	Set for the other PCU states that are reachable out of debug: *Program*, *Data*, *Change of Flow*, *Error in Loop*, *Debug*, *Functional Unit*, *Save,* or *Restore*.
DEBUG	on	on	on	on	on	on	on	Set regardless of the PCU state when clock gating is turned off to use the OnCE features (either *clk_gating_off* pin high or ONCE_ENB[0] set).

1 *off*: no clock

2 *wait*: only clocked when accessed or stimulated

3 *on*: clock is always running

It is possible to control the SDMA power mode. The procedures to force the SDMA into either mode are described in Section 52.9.1.3.1, SLEEP Mode."

52.9.1.3.1 SLEEP Mode

This is the default mode after reset; therefore, resetting the SDMA forces this mode. However, the common procedure is as follows:

- Ensure the *clk_gating_off* pin is low and ONCE_ENB[0] is cleared.
- Disable all channels (via the STOP_STAT control register, and the HO, DO, EO if necessary).
- Wait for the active channels to complete or force a reschedule via the reschedule bit in the RESET register.
- The SDMA is in SLEEP mode making it possible to completely shut off its clock from the chip level clock controller using the procedure described in Section 52.9.1.4, Stop Mode Response."

52.9.1.3.2 RUN Mode

This is the default mode when a channel is running:

- Ensure the *clk_gating_off* pin is low and ONCE_ENB[0] is cleared.
- Activate at least one channel (via the HSTART control registers, a DMA request, and/or the HO, DO, EO register bits).

52.9.1.3.3 DEBUG Mode

The DEBUG mode must be set when one needs to use the debugging facilities of the SDMA:

- Ensure the SDMA clocks are running from the CCM.
- Set the *clk_gating_off* pin high or use the SDMA to set ONCE_ENB[0].

52.9.1.4 Stop Mode Response

The SDMA receives a stop request from the chip level clock controller. This request may be asserted when the chip enters the stop low power mode. If the SDMA is running when the request is received, then the SDMA will complete all pending channels before returning to the SLEEP state. The SDMA sends an acknowledgement to the clock controller when the SLEEP state is entered indicating that the SDMAs clocks can be turned off.

52.9.2 Reset

After reset (either received from the reset module or a software reset required by the AP), the SDMA is in IDLE mode. It will start its boot code located at address 0 once a channel is activated. Activating a channel can be done by the AP after programming a positive priority and setting the channel bit in the EVTPEND register.

There will not be a context RESTORE for the first channel (bootload channel) called after a reset because the context data in RAM has not been initialized. Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized. Subsequent calls to the same channel or different channels may use any of the dynamic context modes

52.10 Software Interface

A separate document exists that fully describes the SDMA Application Programming Interface (API): Refer to the latest revision of document *i.MX51 SDMA Scripts User Manual.*

52.11 Initialization Information

This section discusses the following:

- Section 52.11.1, Hardware Reset”
- Section 52.11.2, Channel Script Execution”
- Section 52.11.3, Initialization and Script Execution Setup Sequence”

52.11.1 Hardware Reset

After reset, the program RAM, context RAM, data RAM, and RAM containing the channel enable registers (CHNENBLn) have unpredictable contents. The active register set is assigned to channel 0 and the PC is initialized to all zeros. However, since the channel enable register is all zeros, there are no active channels and the SDMA is halted waiting for the boot channel to start.

The AP will have to setup the SDMA in order to boot it. The CONFIG register must be initialized to determine the DMA/core clock ratio (1 or 2). and whether the Burst DMA2 unit is used. Channel Enable Registers must also be initialized.

To start up the SDMA, the AP first creates some channel control blocks (CCB) and buffer descriptors (BD) in AP memory for the boot channel (channel 0) and then initializes the channel 0 pointer register (MC0PTR) to the address of the first control block. Section 52.23.1, Data Structures for Boot Code and Channel Scripts” provides an overview of the data structure for the CCB and BD’s. The HSTART,

HOSTOVR and EVTOVR registers are then configured according to Equation 52-2 on page 52-17 to allow channel 0 to run.

Upon being enabled, the SDMA begins executing the script located at the address indicated by the Channel 0 Boot Address register (CHN0ADDR) in the program memory. The reset value of CHN0ADDR points to the default bootload script in ROM. This ROM script will read the channel 0 pointer register (MC0PTR) to determine the location of the Channel Control Block (CCB) in AP memory. The script will then begin fetching by DMA the first channel control block which contains a pointer to the location channel 0 Buffer Descriptor chain which is also fetched via DMA. If the buffer descriptor contains a valid command, the script interprets the command in each buffer descriptor and proceeds to implement the command and move on to the next buffer descriptor control block. The buffer descriptor commands for channel zero are typically set up to load SDMA's program RAM, Data RAM, and initial values for the channel contexts. Some channel scripts expect particular parameters to be passed

There are two ways to make the SDMA boot on a user-defined script. The OnCE (either via its JTAG interface or its AP Control interface) can be used to download any code in the SDMA RAM and force the SDMA to boot on that code. Also, the CHN0ADDR register in the AP programming model can be modified to point to user code in RAM which would need to either have been loaded via the ONCE or default bootload routine (ex before a S/W reset).

52.11.2 Channel Script Execution

The execution of an SDMA script depends on both the instructions that make up the script, the data context upon which it operates, and commands or parameters passed in the buffer descriptor or. All these items must be initialized before the script is allowed to execute. Each of the 32 channels has a separate context, but may share scripts and locations in data RAM.

Channels are initialized by the AP by using channel 0 to download any required scripts and data values and the channels initial context. The context contains all the initial values of the SDMA core registers. This includes the Program Counter (PC) which is set to the start of the desired script in SDMA program memory.

The AP selects which trigger conditions that must occur for the channel to start by configuring the CHNENBL, HOSTOVR and EVTOVR registers. The trigger events include AP setting HE (HSTART) or a hardware DMA request asserts an event input to SDMA. The channel can becomes active according to its priority compared with other runnable channels when the selected trigger(s) cause Equation 52-2 on page 52-17 to evaluate as true.

The specific parameters to be passed to each script in the buffer descriptor or context are documented in the software documentation for each script. Please refer to Section 52.25, References" for complete script documentation. Section 52.23.1.1, Buffer Descriptor Format" provides an overview of the buffer descriptor format.

52.11.3 Initialization and Script Execution Setup Sequence

To summarize, the following steps are minimally required to setup SDMA and run channel scripts.

- Perform Hardware Reset. The program RAM, context RAM, data RAM and CHNENBLn registers have unpredictable contents after this reset.
- Initialize CHNENBLn registers to map DMA request events to desired channels.
- Configure CHNPRIn registers to select priority for runnable channels, A non-zero priority is required for the channel to run.
- Configure the CONFIG register to select DMA to SDMA core clock ratio.
- Set up channel control blocks and buffer descriptors in AP to specify the loading of SDMA program RAM and channel contexts for each SDMA channel to be used. Reference Section 52.23.1, Data Structures for Boot Code and Channel Scripts."
- Configure MC0PTR register with base address of AP Channel Control Block base address.
- Initialize CHNENBLn registers to map DMA request events to associated channel. Reference Section 52.4.3.3, Mapping DMA Requests to Pending Channels."
- Configure CHNPRIn registers to set priority for each channel to be run.
- For each channel to be run, configure HOSTOVR (HO) and EVTOVR (EO) registers to select which events (hardware and/or software trigger events) must occur for the channel to be runnable. Reference Section 52.4.3.5, Runnable Channels Evaluation."
- Set bit 0 of the HSTART register to set HE[0] and allow Channel 0 to run (assumes EO[0] andDO[0] were both set in previous step). This will cause SDMA to load the program RAM and channel contexts configured previously.
- Wait for Channel 0 to finish running. This is indicated by HI[0]=1 in the SDMA_INTR register, or by optional interrupt to the AP.
- Set the LOCK bit in the SDMA_LOCK register to prevent un-authorized uploads of data to SDMA RAM.
- Additional channel scrips can now be run by enabling the selected software or hardware trigger event according to Equation 52-2.

52.12 AP Memory Map and Control Register Definitions

The AP controls the SDMA by means of several interface registers. Those registers are described in the current section.

52.12.1 AP Memory Map

The following table provides the memory map for the AP control SDMA registers.

Table 52-9. AP Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0x83FD_4000 (MC0PTR)	AP (MCU) Channel 0 Pointer	R/W	0x0000_0000	52.12.3.2/52-49
0x83FD_4004 (INTR)	Channel Interrupts	R/W	0x0000_0000	52.12.3.2/52-49
0x83FD_4008 (STOP_STAT)	Channel Stop/Channel Status	R	0x0000_0000	52.12.3.3/52-50

Table 52-9. AP Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0x83FD_400C (HSTART)	Channel Start	R/W	0x0000_0000	52.12.3.4/52-50
0x83FD_4010 (EVTOVR)	Channel Event Override	R/W	0x0000_0000	52.12.3.5/52-51
0x83FD_4014 (DSPOVR)	Channel BP Override	R/W	0xFFFF_FFFF	52.12.3.6/52-52
0x83FD_4018 (HOSTOVR)	Channel AP Override	R/W	0x0000_0000	52.12.3.7/52-53
0x83FD_401C (EVTPEND)	Channel Event Pending	R	0x0000_0000	52.12.3.8/52-53
0x83FD_4024 (RESET)	Reset Register	R	0x0000_0000	52.12.3.9/52-54
0x83FD_4028 (EVTERR)	DMA Request Error Register	R	0x0000_0000	52.12.3.10/52-55
0x83FD_402C (INTRMASK)	Channel AP Interrupt Mask	R/W	0x0000_0000	52.12.3.11/52-56
0x83FD_4030 (PSW)	Schedule Status	R	0x0000_0000	52.12.3.12/52-57
0x83FD_4034 (EVTERRDBG)	DMA Request Error Register	R	0x0000_0000	52.12.3.13/52-58
0x83FD_4038 (CONFIG)	Configuration Register	R/W	0x0000_0003	52.12.3.14/52-59
0x83FD_403C (SDMA_LOCK)	SDMA LOCK	R/W	0x0000_0000	52.12.3.15/52-60
0x83FD_4040 (ONCE_ENB)	OnCE Enable	R/W	0x0000_0000	52.12.3.16/52-61
0x83FD_4044 (ONCE_DATA)	OnCE Data Register	R/W	0x0000_0000	52.12.3.17/52-62
0x83FD_4048 (ONCE_INSTR)	OnCE Instruction Register	R/W	0x0000_0000	52.12.3.18/52-63
0x83FD_404C (ONCE_STAT)	OnCE Status Register	R	0x0000_E000	52.12.3.19/52-63
0x83FD_4050 (ONCE_CMD)	OnCE Command Register	R/W	0x0000_0000	52.12.3.20/52-65
0x83FD_4058 (ILLINSTADDR)	Illegal Instruction Trap Address	R/W	0x0000_0001	52.12.3.21/52-66
0x83FD_405C (CHN0ADDR)	Channel 0 Boot Address	R/W	0x0000_0050	52.12.3.22/52-66
0x83FD_4060 (EVT_MIRROR)	DMA Requests	R	0x0000_0000	52.12.3.23/52-67
0x83FD_4064 (EVT_MIRROR2)	DMA Requests 2	R	0x0000_0000	52.12.3.24/52-68
0x83FD_4070 (XTRIG_CONF1)	Cross-Trigger Events Configuration Register 1	R/W	0x0000_0000	52.12.3.25/52-69
0x83FD_4074 (XTRIG_CONF2)	Cross-Trigger Events Configuration Register 2	R/W	0x0000_0000	52.12.3.25/52-69
0x83FD_40x078 (OTB)	Once Trace Buffer Register	R	0x0000_0000	52.12.3.26/52-71
0x83FD_40x07C (PRF_CNT_1) to 0x83FD_40x090 (PRF_CNT_6)	Profile counter Registers	R/W	0x0000_0000	52.12.3.27/52-72
0x83FD_40x094 (PRF_CFG)	Profile config/status Register	R/W	0x0000_0000	52.12.3.28/52-73

Table 52-9. AP Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0x83FD_4100+n*4 (CHNPRIn)[1]	Channel Priority Registers	R/W	0x0000_0000	52.12.3.29/52-75
0x83FD_4200+n*4 (CHNENBLn)[2]	Channel Enable RAM	R/W	Undefined	52.12.3.30/52-76
0x83FD_42C0 - 0x83FD_42FC	Reserved	-	Undefined	

[1] CHNPRIn: For n= 0 to 31

[2] CHNENBLn: For n= 0 to 47

52.12.2 Register Summary

The following definitions serve as a key for the AP control SDMA register summary.

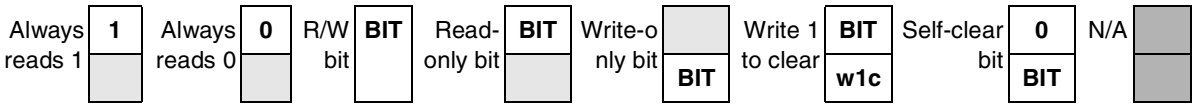


Figure 52-11. Key to Register Fields

Table 52-10 provides a key for register figures.

Table 52-10. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

All registers are clocked with the SDMA clock (which means the AP must ensure that the SDMA clock is running when it wants to access any register).

Table 52-11. AP SDMA Register Summary

Name		31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0x83FD_4000 (MC0PTR)	R	MCOPTR[31:16]
	W	
	R	MCOPTR[15:0]
	W	
0x83FD_4004 (INTR)	R	HI[31:16]
	W	w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c
	R	HI[15:0]
	W	w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c
0x83FD_4008 (STOP_STAT)	R	HE[31:16]
	W	w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c
	R	HE[15:0]
	W	w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c w1c
0x83FD_400C (HSTART)	R	HSTART[31:16]/HE[31:16]
	W	slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr
	R	HSTART[15:0]/HE[15:0]
	W	slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr
0x83FD_4010 (EVTOVR)	R	EO[31:16]
	W	
	R	EO[15:0]
	W	
0x83FD_4014 (DSPOVR)	R	DO[31:16]
	W	
	R	DO[15:0]
	W	
0x83FD_4018 (HOSTOVR)	R	HO[31:16]
	W	
	R	HO[15:0]
	W	

Table 52-11. AP SDMA Register Summary (continued)

Name		31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0x83FD_401C (EVTPEND)	R	EP[31:16]
	W	slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr
	R	EP[15:0]
	W	slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr slfclr
0x83FD_4024 (RESET)	R	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	W	
	R	0 0 0 0 0 0 0 0 0 0 0 0 0 0 RESCHED RESET
	W	
0x83FD_4028 (EVTERR)	R	CHNERR[31:16]
	W	
	R	CHNERR[15:0]
	W	
0x83FD_402C (INTRMASK)	R	HIMASK[31:16]
	W	
	R	HIMASK[15:0]
	W	
0x83FD_4030 (PSW)	R	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	W	
	R	NCP[2:0] NCR[4:0] CCP[2:0] CCR[4:0]
	W	
0x83FD_4034 (EVTERRDBG)	R	CHNERR[31:16]
	W	
	R	CHNERR[15:0]
	W	
0x83FD_4038 (CONFIG)	R	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
	W	
	R	0 0 0 DSP DMA RTD OBS 0 0 0 0 0 0 ACR 0 0 CSM[1:0]
	W	

Table 52-11. AP SDMA Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x83FD_403C (SDMA_LOCK)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRESET LOCK CLR	LOCK
	W																
0x83FD_4040 (ONCE_ENB)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ENB
	W																
0x83FD_4044 (ONCE_DATA)	R	DATA[31:16]															
	W																
	R	DATA[15:0]															
	W																
0x83FD_4048 (ONCE_INSTR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	INSTRUCTION[15:0]															
	W																
0x83FD_404C (ONCE_STAT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0	0	0	0	ECDR[2:0]		
	W																
0x83FD_4050 (ONCE_CMD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	CMD[3:0]			
	W																
0x83FD_4058 (ILLINSTADDR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	ILLINSTADDR[13:0]													
	W																

Table 52-11. AP SDMA Register Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x83FD_405C (CHN0ADDR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	SMSZ	CHN0ADDR[13:0]													
	W																
0x83FD_4060 (EVT_MIRROR)	R	EVENTS[31:16]															
	W																
	R	EVENTS[15:0]															
	W																
0x83FD_4064 (EVT_MIRROR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	EVENTS[47:32]															
	W																
0x83FD_4070 (XTRIG_CONF1)	R	0	CNF3	NUM3						0	CNF 2	NUM2					
	W																
	R	0	CNF1	NUM1						0	CNF 0	NUM0					
	W																
0x83FD_4074 (XTRIG_CONF2)	R	0	CNF7	NUM7						0	CNF 6	NUM6					
	W																
	R	0	CNF5	NUM5						0	CNF 4	NUM4					
	W																
0x83FD_40x078 (OTB)	R	0	0	0	TBF	TADDR[13:2]											
	W																
	R	TADDR[1:0]		CHFADDR[13:0]													
	W																

Table 52-11. AP SDMA Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x83FD_40x07C (PRF_CNT_1) to 0x83FD_40x090 (PRF_CNT_6)	R	COUNTER_CONFIG									OFL	COUNTER					
	W																
	R	COUNTER															
	W																
0x83FD_40x094 (PRF_CFG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	ISR	OFL 6	OFL 5	OFL 4	OFL 3	OFL 2	OFL 1	INT_ EN_ 6	INT_ EN_ 5	INT_ EN_ 4	INT_ EN_ 3	INT_ EN_ 2	INT_ EN_ 1	EN
	W			w1c													
0x83FD_4100+n*4 (CHNPRIn)[1]	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	CHNPRIn[2:0]		
	W																
0x83FD_4200+n*4 (CHNENBLn)[2]	R	ENBLn[31:16]															
	W																
	R	ENBLn[15:0]															
	W																
0x83FD_42C0-0x83FD_42FC RESERVED	R	RESERVED															
	W																
	R																
	W																

1 CHNPRIn: For n=0 to 31

2 CHNENBLn: For n=0 to 47

52.12.3 Register Descriptions

The following sections provide figures and detailed field descriptions of the SDMA registers.

52.12.3.1 AP Channel 0 Pointer (MC0PTR)

The following table presents the register; Table 52-12 provides its field descriptions.

0x83FD_4000 (MC0PTR) Access: User Read/Write
Wait State: 0

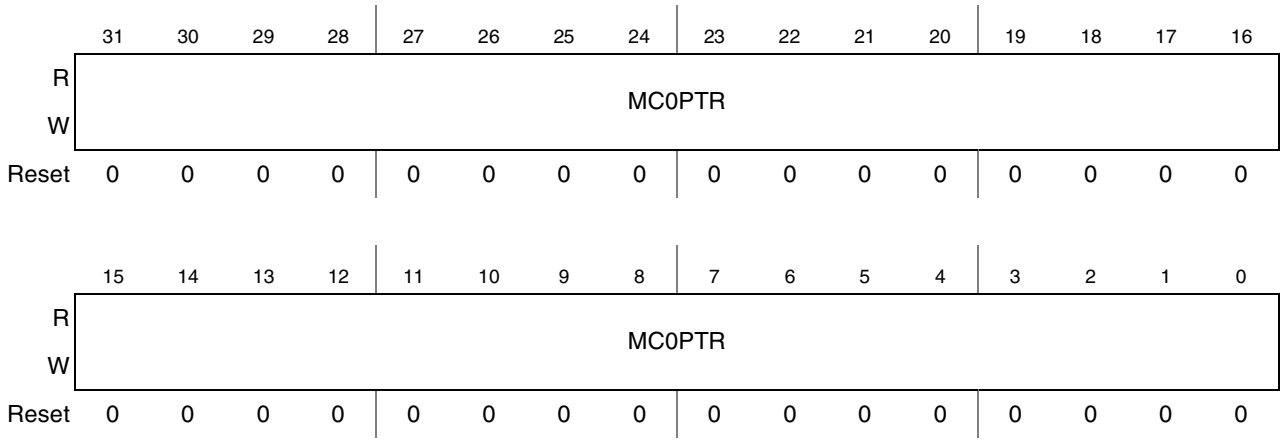


Figure 52-12. AP Channel 0 Pointer (MC0PTR)

Table 52-12. MC0PTR Field Descriptions

Field	Description
31–0 MC0PTR	**Channel 0 Pointer** contains the 32-bit address, in AP memory, of channel 0 control block (the boot channel). Refer to the document *i.MX51 SDMA Scripts User Manual* for the use of this register. The AP has a read/write access and the SDMA has a read-only access.

52.12.3.2 Channel Interrupts (INTR)

Figure 52-13 presents the register; Table 52-13 provides its field descriptions.

0x83FD_4004 (INTR) Access: User Read/Write
Wait State: 0

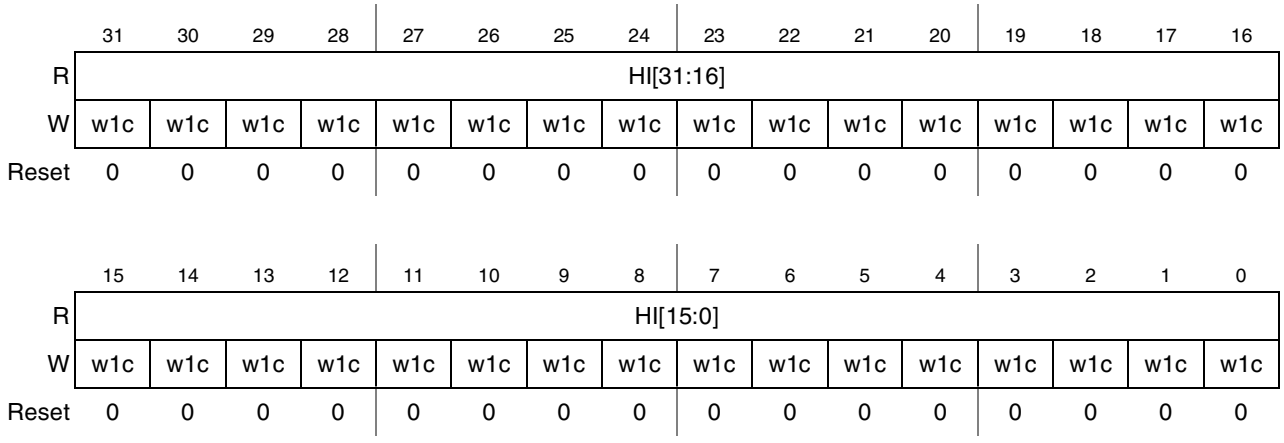


Figure 52-13. Channel Interrupts (INTR) Register

Table 52-13. INTR Field Descriptions

Field	Description
31–0 HI[31:0]	The AP Interrupts register contains the 32 HI[i] bits. If any bit is set, it will cause an interrupt to the AP. This register is a “write-ones” register to the AP. When the AP sets a bit in this register the corresponding HI[i] bit is cleared. The interrupt service routine should clear individual channel bits when their interrupts are serviced, failure to do so will cause continuous interrupts. The SDMA is responsible for setting the HI[i] bit corresponding to the current channel when the corresponding done instruction is executed.

52.12.3.3 Channel Stop/Channel Status (STOP_STAT)

Figure 52-14 presents the register; Table 52-14 provides its field descriptions.

0x83FD_4008 (STOP_STAT) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HE[31:16]															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HE[15:0]															
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-14. Channel Stop/Channel Status (STOP_STAT) Register

Table 52-14. STOP_STAT Field Descriptions

Field	Description
31–0 HE	This 32-bit register gives access to the AP Enable bits. There is one bit for every channel. This register is a “write-ones” register to the AP. When the AP writes 1 in bit i of this register, it clears the HE[i] and HSTART[i] bits. Reading this register yields the current state of the HE[i] bits.

52.12.3.4 Channel Start (HSTART)

Figure 52-15 presents the register; Table 52-15 provides its field descriptions.

0x83FD_400C (HSTART) Access: User Read-Only
Wait State: 0

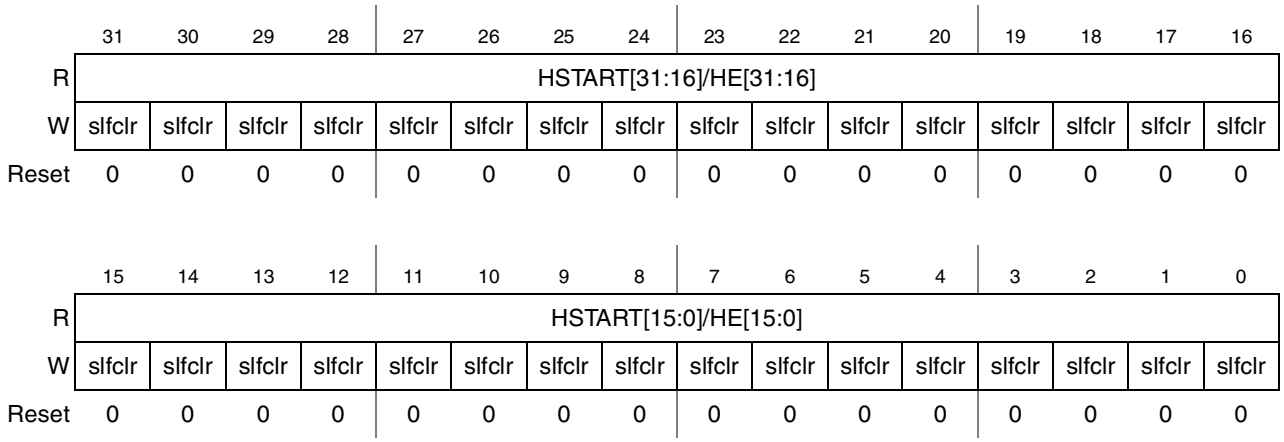


Figure 52-15. Channel Start (HSTART) Register

Table 52-15. HSTART Field Descriptions

Field	Description
31–0 HSTART/HE	The HSTART/HE registers are 32 bits wide with one bit for every channel. When a bit is written to 1, it enables the corresponding channel. Two physical registers are accessed with that address (HSTART and HE), which enables the AP to trigger a channel a second time before the first trigger is processed. • This register is a “write-ones” register to the AP. Neither HSTART[i] bit can be set while the corresponding HE[i] bit is cleared. • When the AP tries to set the HSTART[i] bit by writing a one (if the corresponding HE[i] bit is clear), the bit in the HSTART[i] register will remain cleared and the HE[i] bit will be set. • If the corresponding HE[i] bit was already set, the HSTART[i] bit will be set. The next time the SDMA channel *i* attempts to clear the HE[i] bit by means of a done instruction, the bit in the HSTART[i] register will be cleared and the HE[i] bit will take the old value of the HSTART[i] bit. • Reading this register yields the current state of the HSTART[i] bits. This mechanism enables the AP to pipeline two HSTART commands per channel.

52.12.3.5 Channel Event Override (EVTOVR)

Figure 52-16 presents the register; Table 52-16 provides its field descriptions.

0x83FD_4010 (EVTOVR) Access: User Read/Write
Wait State: 0

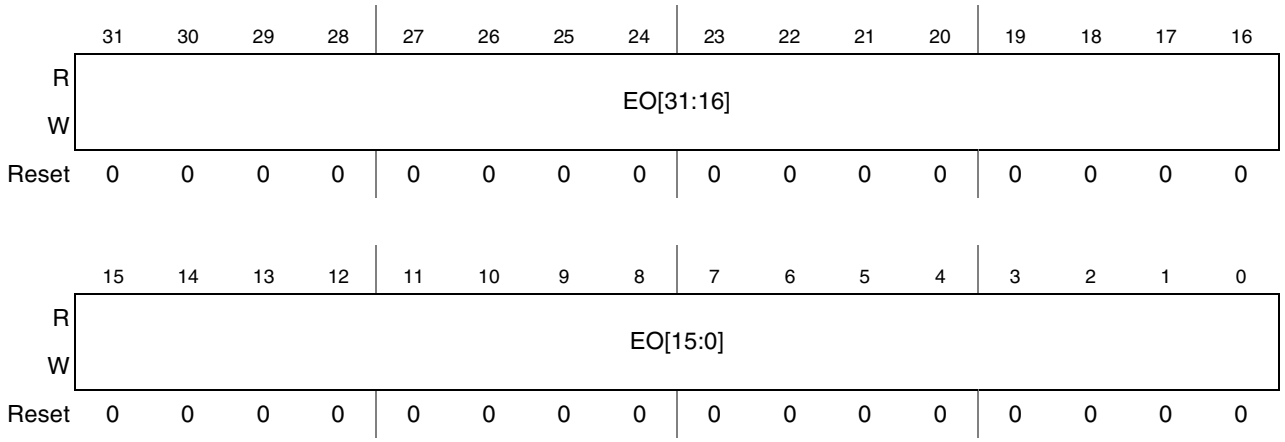


Figure 52-16. Channel Event Override (EVTOVR) Register

Table 52-16. EVTOVR Field Descriptions

Field	Description
31–0 EO	The Channel Event Override register contains the 32 EO[i] bits. A bit set in this register causes the SDMA to ignore DMA requests when scheduling the corresponding channel.

52.12.3.6 Channel BP Override (DSPOVR)

Figure 52-17 presents the register; Table 52-17 provides its field descriptions.

0x83FD_4014 (DSPOVR) Access: User Read/Write
Wait State: 0

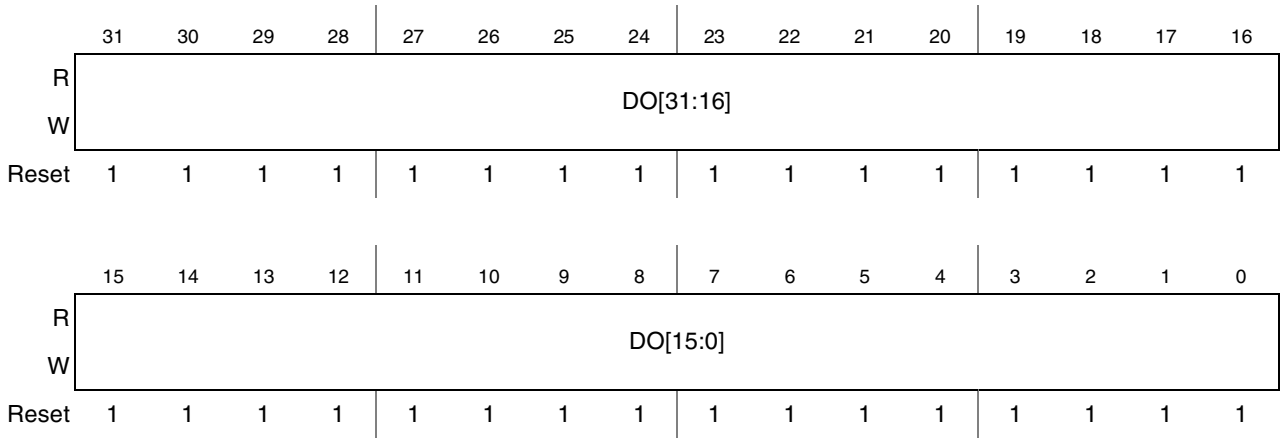


Figure 52-17. Channel DSP Override (DSPOVR) Register

Table 52-17. DSPOVR Field Descriptions

Field	Description
31–0 DO	This register is reserved. **All DO bits should be set to the reset value of 1.** A setting of 0 will prevent SDMA channels from starting according to Equation 52-2 on page 52-17. 0 - Reserved 1 - Reset value.

52.12.3.7 Channel AP Override (HOSTOVR)

Figure 52-18 presents the register; Table 52-18 provides its field descriptions.

0x83FD_4018 (HOSTOVR) Access: User Read/Write
Wait State: 0

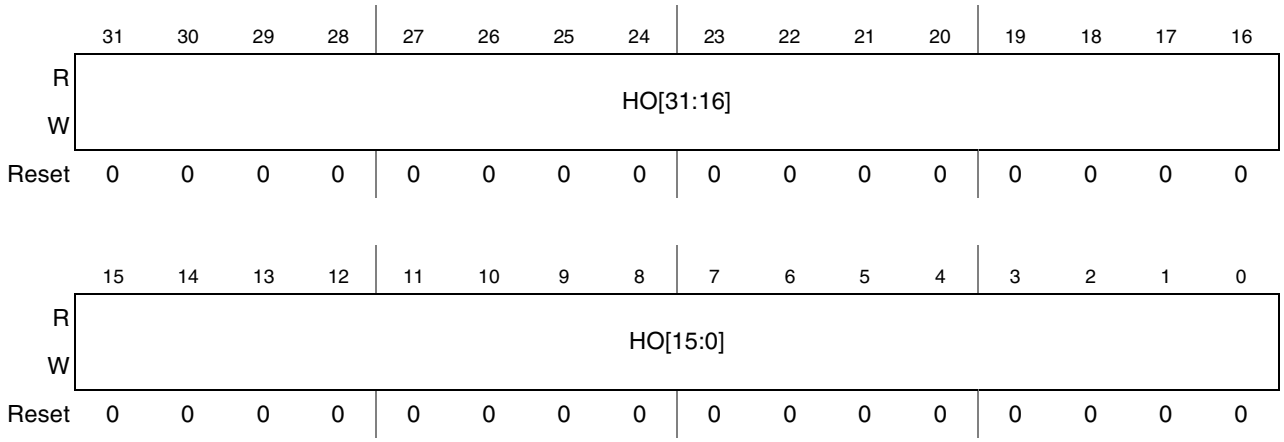


Figure 52-18. Channel AP Override (HOSTOVR) Register

Table 52-18. HOSTOVR Field Descriptions

Field	Description
31–0 HO	The Channel AP Override register contains the 32 HO[i] bits. A bit set in this register causes the SDMA to ignore the AP enable bit (HE) when scheduling the corresponding channel.

52.12.3.8 Channel Event Pending (EVTPEND)

Figure 52-19 presents the register; Table 52-19 provides its field descriptions.

0x83FD_401C (EVTPEND) Access: User Read-Only
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EP[31:16]															
W	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EP[15:0]															
W	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr	slfclr
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-19. Channel Event Pending (EVTPEND) Register

Table 52-19. EVTPEND Field Descriptions

Field	Description
31–0 EP	The Channel Event Pending register contains the 32 EP[i] bits. Reading this register enables the AP to determine what channels are pending after the reception of a DMA request. • Setting a bit in this register causes the SDMA to reevaluate scheduling as if a DMA request mapped on this channel had occurred. This is useful for starting up channels, so that initialization is done before awaiting the first request. The scheduler can also set bits in the EVTPEND register according to the received DMA requests. • The EP[i] bit may be cleared by the done instruction when running the channel *i* script. This a “write-ones” mechanism: Writing a ‘0’ does not clear the corresponding bit.

52.12.3.9 Reset Register (RESET)

Figure 52-20 presents the register; Table 52-20 provides its field descriptions.

0x83FD_4024 (RESET) Access: User Read-Only
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RESCHED	RESET
W															slfclr	slfclr
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-20. Reset Register

Table 52-20. RESET Field Descriptions

Field	Description
31–2	Reserved
1 RESCHED	When set, this bit forces the SDMA to reschedule as if a script had executed a done instruction. This enables the AP to recover from a runaway script on a channel by clearing its HE[i] bit via the STOP register, and then forcing a reschedule via the RESCHED bit. The RESCHED bit is cleared when the context switch starts.
0 RESET	When set, this bit causes the SDMA to be held in a software reset. The internal reset signal is held low 16 cycles; the RESET bit is automatically cleared when the internal reset signal rises.

52.12.3.10 DMA Request Error Register (EVTERR)

Figure 52-21 presents the register; Table 52-21 provides its field descriptions.

0x83FD_4028 (EVTERR) Access: User Read-Only
Wait State: 0

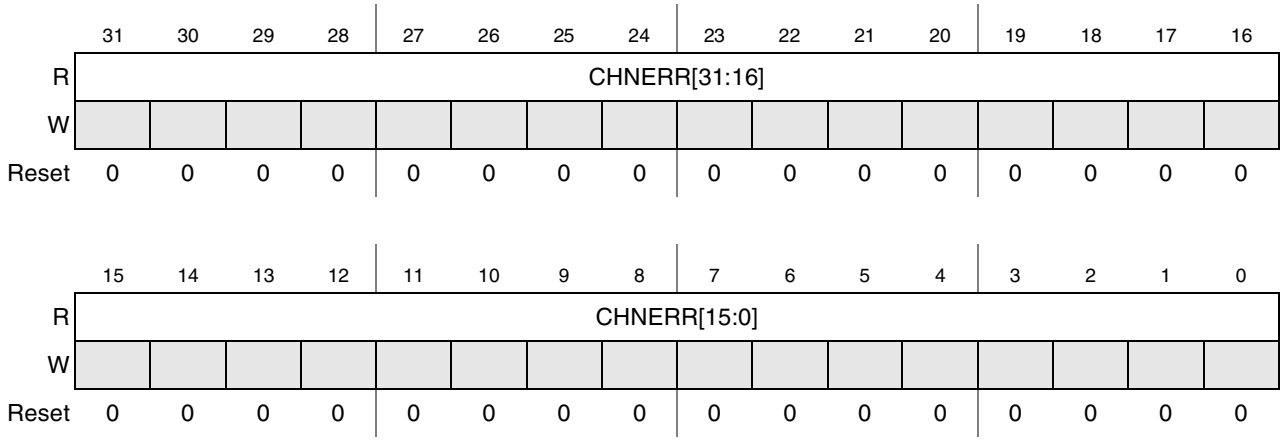


Figure 52-21. DMA Request Error (EVTERR) Register

Table 52-21. EVTERR Field Descriptions

Field	Description
31–0 CHNERR	This register is used by the SDMA to warn the AP when an incoming DMA request was detected and it triggers a channel that is already pending or being serviced. This probably means there is an overflow of data for that channel. • An interrupt is sent to the AP if the corresponding channel bit is set in the INTRMASK register. • This is a “write-ones” register for the scheduler. It is only able to set the flags. The flags are cleared when the register is read by the AP or during SDMA reset. • The CHNERR[i] bit is set when a DMA request that triggers channel *i* is received through the corresponding input pins and the EP[i] bit is already set; the EVTERR[i] bit is unaffected if the AP tries to set the EP[i] bit, whereas, that EP[i] bit is already set.

52.12.3.11 Channel AP Interrupt Mask Flags (INTRMASK)

Figure 52-22 presents the register; Table 52-22 provides its field descriptions.

0x83FD_402C (INTRMASK) Access: User Read/Write
Wait State: 0

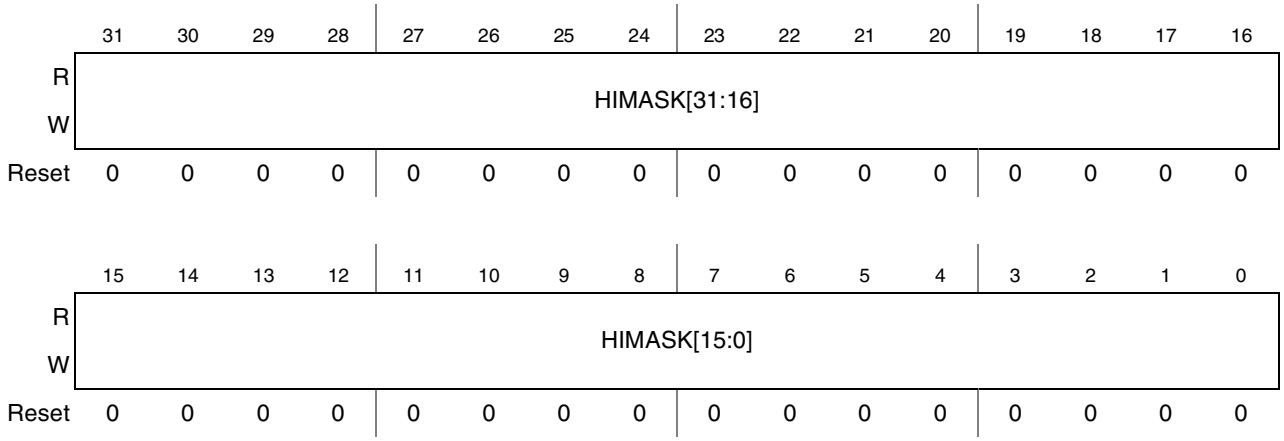


Figure 52-22. Channel AP Interrupt Mask Flags (INTRMASK) Register

Table 52-22. INTRMASK Field Description

Field	Description
31–0 HIMASK	The Interrupt Mask Register contains 32 interrupt generation mask bits. If bit HIMASK[i] is set, the HI[i] bit is set and an interrupt is sent to the AP when a DMA request error is detected on channel *i* (for example, EVTERR[i] is set).

52.12.3.12 Schedule Status (PSW)

Figure 52-23 presents the register; Table 52-23 provides its field descriptions.

0x83FD_4030 (PSW) Access: User Read-Only
Wait State: 0

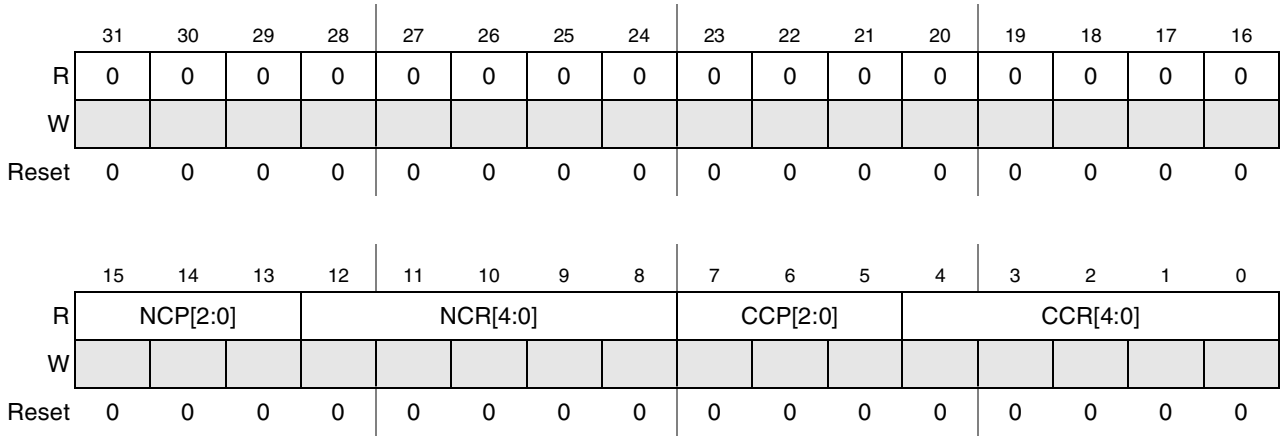


Figure 52-23. Schedule Status (PSW) Register

Table 52-23. PSW Field Descriptions

Field	Description
31–16	Reserved
15–13 NCP[2:0]	The Next Channel Priority gives the next pending channel priority. When the priority is 0, it means there is no pending channel and the NCR value has no meaning. 0 No running channel 1–7 Active channel priority
12–8 NCR[4:0]	The Next Channel Register indicates the number of the next scheduled pending channel with the highest priority.
7–4 CCP[2:0]	The Current Channel Priority indicates the priority of the current active channel. When the priority is 0, no channel is running: The SDMA is idle and the CCR value has no meaning. In the case that the SDMA has finished running the channel and has entered sleep state, CCP will indicate the priority of previous running channel. 0 No running channel 1–7 Active channel priority
3–0 CCR[4:0]	The Current Channel Register indicates the number of the channel that is being executed by the SDMA.In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel.

52.12.3.13 DMA Request Error Register for Debug (EVTERRDBG)

Figure 52-24 presents the register; Table 52-24 provides its field descriptions.

0x83FD_4034 (EVTERRDBG)
Wait State: 0

Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHNERR[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-24. DMA Request Error for Debug (EVTERRDBG) Register

Table 52-24. EVTERRDBG Field Descriptions

Field	Description
31–0 CHNERR	This register is the same as EVTERR, except reading it does not clear its contents. This address is meant to be used in debug mode. The AP OnCE may check this register value without modifying it.

52.12.3.14 Configuration Register (CONFIG)

Figure 52-25 presents the register; Table 52-25 provides its field descriptions.

0x83FD_4038 (CONFIG) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	DSP Ctrl	RTD OBS	0	0	0	0	0	0	ACR	0	0	CSM[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Figure 52-25. Configuration Register (CONFIG)

Table 52-25. CONFIG Register Field Descriptions

Field	Description
31–13	Reserved
12 DSPCtrl	This bit **should be configured as 1.** This implies SDMA is controlled by single core. 0- Dual core to control SDMA 1- Single core to control SDMA
11 RTDOBS	Indicates if Real-Time Debug pins are used: They do not toggle by default in order to reduce power consumption. 0 RTD pins disabled 1 RTD pins enabled
10–6	Reserved
4 ACR	AP DMA / SDMA Core Clock Ratio. Selects the clock ratio between AP DMA interfaces (burst DMA and burst DMA2) and the internal SDMA core clock. The frequency selection is determined separately by the chip clock controller. This bit has to match the configuration of the chip clock controller that generates the clocks used in the SDMA. 0 AP DMA interface frequency equals twice core frequency 1 AP DMA interface frequency equals core frequency

Table 52-25. CONFIG Register Field Descriptions (continued)

Field	Description
3–2	Reserved
1–0 CSM	Selects the Context Switch Mode. The AP has a read/write access. The SDMA cannot modify that register. The value at reset is 3, which selects the dynamic context switch by default. That register can be modified at anytime but the new context switch configuration will only be taken into account at the start of the next restore phase. NOTE: The first call to SDMA's channel 0 Bootload script after reset should use static context switch mode to ensure the context RAM for channel 0 is initialized in the channel SAVE Phase. After Channel 0 is run once, then any of the dynamic context modes can be used. 0 static 1 dynamic low power 2 dynamic with no loop 3 dynamic

52.12.3.15 SDMA Lock Register (SDMA_LOCK)

Figure 52-26 presents the register; Table 52-26 provides its field descriptions.

0x83FD_403C (SDMA_LOCK) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRESET LOCK CLR	LOCK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-26. SDMA LOCK(SDMA_LOCK) Register

Table 52-26. ONCE_ENB Field Descriptions

Field	Description
31–2	Reserved
1 SRESET_LOCK_CLR	The SRESET_LOCK_CLR bit determine if the LOCK bit is cleared on a software reset triggered by writing to the RESET register. This bit cannot be changed if LOCK=1. SREST_LOCK_CLR is cleared by conditions that clear the LOCK bit. 0 Software Reset does not clear the LOCK bit. 1 Software Reset clears the LOCK bit.
0 LOCK	The LOCK bit is used to restrict access to update SDMA script memory through ROM channel zero scripts and through the OnCE interface under MCU control. The LOCK bit is set: • The SDMA_LOCK, ONCE_ENB,CH0ADDR, and ILLINSTADDR registers cannot be written. These registers can be read, but writes are ignored. • SDMA software executing out of ROM or RAM may check the LOCK bit in the LOCK register on page 52-106 to determine if certain operations are allowed, such as up-loading new scripts. Once the LOCK bit is set to 1, only a reset can clear it. The LOCK bit is cleared by a hardware reset. LOCK is cleared by a software reset only if SRESET_LOCK_CLR is set. 0 LOCK disengaged. 1 LOCK enabled.

52.12.3.16 OnCE Enable (ONCE_ENB)

Figure 52-27 presents the register; Table 52-27 provides its field descriptions.

0x83FD_4040 (ONCE_ENB) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ENB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-27. OnCE Enable (ONCE_ENB) Register

Table 52-27. ONCE_ENB Field Descriptions

Field	Description
31–1	Reserved
0 ENB	The OnCE Enable register selects the OnCE control source: When cleared (0), the OnCE registers are accessed through the JTAG interface; when set (1), the OnCE registers may be accessed by the AP through the addresses described, as follows. • After reset, the OnCE registers are accessed through the JTAG interface. • Writing a 1 to ENB enables the AP to access the ONCE_* as any other SDMA control register. • When cleared (0), all the ONCE_xxx registers cannot be written. The value of ENB cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

52.12.3.17 OnCE Data Register (ONCE_DATA)

Figure 52-28 presents the register; Table 52-28 provides its field descriptions.

0x83FD_4044 (ONCE_DATA) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-28. OnCE Data Register (ONCE_DATA)

Table 52-28. ONCE_DATA Field Descriptions

Field	Description
31–0 DATA	Data register of the OnCE JTAG controller. Refer to Section 52.18.3, OnCE and Real-Time Debug” for information on this register.

52.12.3.18 OnCE Instruction Register (ONCE_INSTR)

Figure 52-29 presents the register; Table 52-29 provides its field descriptions.

0x83FD_4048 (ONCE_INSTR) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INSTR[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-29. OnCE Instruction Register (ONCE_INSTR)

Table 52-29. ONCE_INSTR Field Descriptions

Field	Description
31–16	Reserved
15–0 INSTR	Instruction register of the OnCE JTAG controller. Refer to Section 52.18.3, OnCE and Real-Time Debug" for information on this register.

52.12.3.19 OnCE Status Register (ONCE_STAT)

Figure 52-30 presents the register; Table 52-30 provides its field descriptions.

0x83FD_404C (ONCE_STAT) Access: User Read-Only
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0	0	0	0	ECDR[2:0]		
W																
Reset	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-30. OnCE Status Register (ONCE_STAT)

Table 52-30. ONCE_STAT Field Descriptions

Field	Description
31–16	Reserved
15–12 PST[3:0]	The Processor Status bits reflect the state of the SDMA RISC engine. Its states are as follows: • The “Program” state is the usual instruction execution cycle. • The “Data” state is inserted when there are wait-states during a load or a store on the data bus (ld or st). • The “Change of Flow” state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions). • The “Change of Flow in Loop” state is used when an error causes a hardware loop exit. • The “Debug” state means the SDMA is in debug mode. • The “Functional Unit” state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In “Sleep” modes, no script is running (this is the RISC engine idle state). The “after Reset” is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation). • The “in Sleep” states are the same as above except they do not have any corresponding channel: They are used when entering debug mode after reset. The reason is that it is necessary to return to the “Sleep after Reset” state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Save 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change Flow in Loop in Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Restore
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an `rbuffer` command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.
7 MST	This flag is raised when the OnCE is controlled from the AP peripheral interface. 0 The JTAG interface controls the OnCE. 1 The AP peripheral interface controls the OnCE.

Table 52-30. ONCE_STAT Field Descriptions (continued)

Field	Description
6–3	Reserved
2–0 ECDR	Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one of the EDR bits is set (the meaning of the encoding is given below). The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addra_cond, addrb_cond, and data_cond conditions. The value of those fields is given by the EDR bits. EDR[0] 1 matched addra_cond EDR[1] 1 matched addrb_cond EDR[2] 1 matched data_cond

52.12.3.20 OnCE Command Register (ONCE_CMD)

Figure 52-31 presents the register; Table 52-31 provides its field descriptions.

0x83FD_4050 (ONCE_CMD) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	CMD[3:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-31. OnCE Command Register (ONCE_CMD)

Table 52-31. ONCE_CMD Field Descriptions

Field	Description
31–4	Reserved
3–0 CMD	Writing to this register will cause the OnCE to execute the command that is written. When needed, the ONCE_DATA and ONCE_INSTR registers should be loaded with the correct value before writing the command to that register. For a list of the OnCE commands and their usage, see Section 52.18.3, OnCE and Real-Time Debug." 0 rstatus 1 dmov 2 exec_once 3 run_core 4 exec_core 5 debug_rqst 6 rbuffer 7–15 *reserved*

52.12.3.21 Illegal Instruction Trap Address (ILLINSTADDR)

Figure 52-32 presents the register; Table 52-32 provides its field descriptions.

0x83FD_4058 (ILLINSTADDR) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	ILLINSTADDR[13:0]													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 52-32. Illegal Instruction Trap Address (ILLINSTADDR)

Table 52-32. ILLINSTADDR Field Descriptions

Field	Description
31–14	Reserved
13–0 ILLINSTADDR	The Illegal Instruction Trap Address is the address where the SDMA jumps when an illegal instruction is executed. It is 0x0001 after reset. The value of ILLINSTADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

52.12.3.22 Channel 0 Boot Address (CHN0ADDR)

Figure 52-33 presents the register; Table 52-33 provides its field descriptions.

0x83FD_405C (CHN0ADDR) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SMSZ	CHN0ADDR[13:0]													
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Figure 52-33. Channel 0 Boot Address (CHN0ADDR) Register

Table 52-33. CHN0ADDR Register Field Descriptions

Field	Description
31–15	Reserved
14 SMSZ	The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context, which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism. 0 24 words per context 1 32 words per context The value of SMSZ cannot be changed if the LOCK bit in the SDMA_LOCK register is set.
13–0 CHN0ADDR	This 14-bit register is used by the boot code of the SDMA. After reset, it points to the standard boot routine in ROM (channel 0 routine). By changing this address, you can perform a boot sequence with your own routine. The very first instructions of the boot code fetch the contents of this register (it is also mapped in the SDMA memory space) and jump to the given address. The reset value is 0x0050 (decimal 80). The value of CHN0ADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

52.12.3.23 DMA Requests (EVT_MIRROR)

Figure 52-34 presents the register; Table 52-34 provides its field descriptions.

0x83FD_4060 (EVT_MIRROR) Access: User Read-Only
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EVENTS[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-34. DMA Requests (EVT_MIRROR)

Table 52-34. EVT_MIRROR Field Descriptions

Field	Description
31–0 EVENTS	This register reflects the DMA requests received by the SDMA for events 31-0. The AP and the SDMA have a read-only access. There is one bit associated with each of 32 DMA request events. This information may be useful during debug of the modules that generate the DMA requests. The EVT_MIRROR register is cleared following read access. 0 DMA request event not pending 1 DMA request event pending

52.12.3.24 DMA Requests 2 (EVT_MIRROR2)

Figure 52-34 presents the register; Table 52-34 provides its field descriptions.

0x83FD_4064 (EVT_MIRROR2) Access: User Read-Only
Wait State: 0

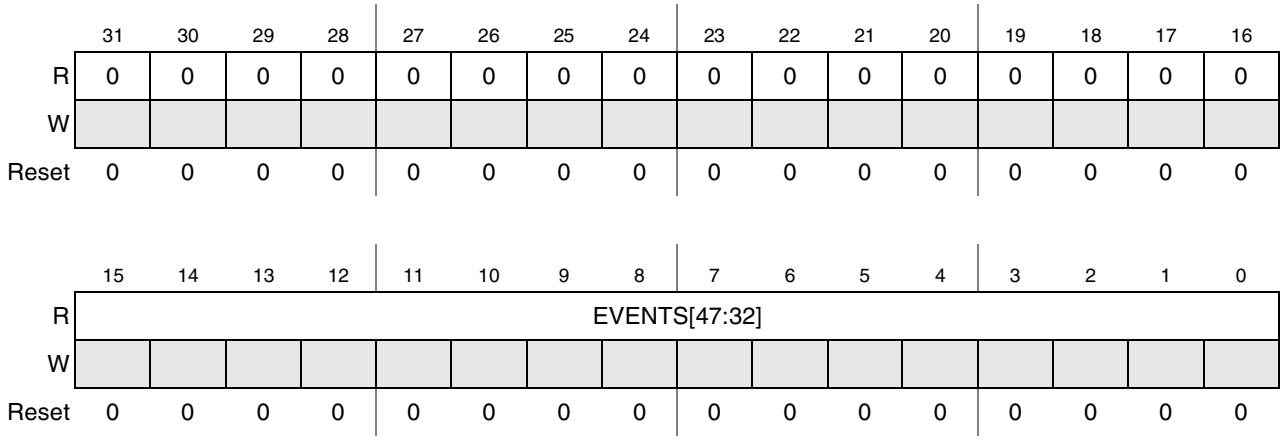


Figure 52-35. DMA Requests (EVT_MIRROR2)

Table 52-35. EVT_MIRROR2 Field Descriptions

Field	Description
31–15	Reserved
15–0 EVENTS[47:32]	This register reflects the DMA requests received by the SDMA for events 47-32. The AP and the SDMA have a read-only access. There is one bit associated with each of DMA request events. This information may be useful during debug of the modules that generate the DMA requests. The EVT_MIRROR2 register is cleared following read access. 0 - DMA request event not pending 1- DMA request event pending

52.12.3.25 Cross-Trigger Events Configuration Register (1) and (2) (XTRIG_CONF1 and XTRIG_CONF2)

Figure 52-36 presents the XTRIG_CONF1 register; Table 52-36 provides its field descriptions.

0x83FD_4070 (XTRIG_CONF1) Access: User Read/Write
Wait State: 0

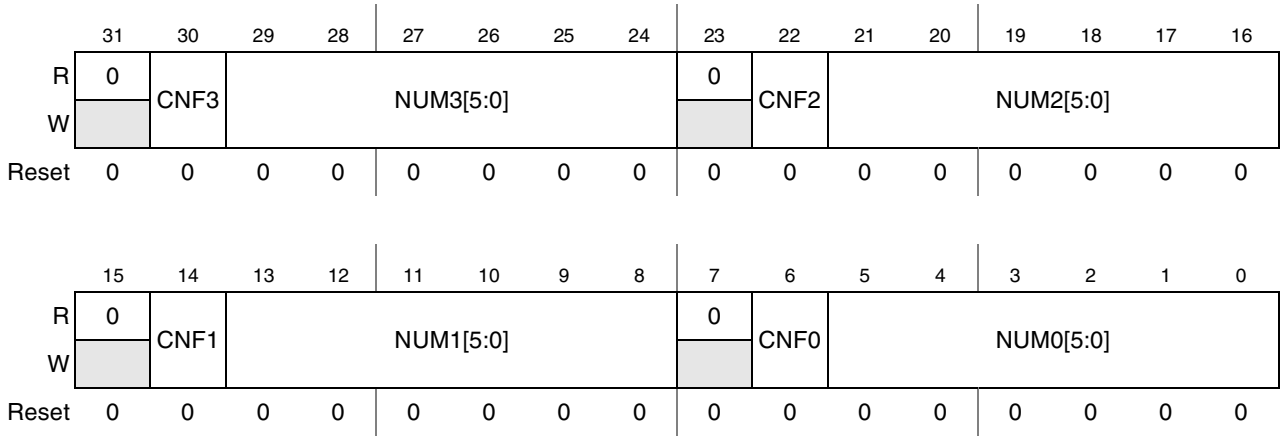


Figure 52-36. Cross-Trigger Events Configuration Register (1) (XTRIG_CONF1)

Table 52-36. XTRIG_CONF1 Field Descriptions

Field	Description
31	Reserved
30 CNF3	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by the reception of a DMA request or by the starting of a channel script execution. 0 channel 1 DMA request
29–24 NUM3[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
23	Reserved
22 CNF2	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
21–16 NUM2[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
15	Reserved
14 CNF1	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request

Table 52-36. XTRIG_CONF1 Field Descriptions (continued)

Field	Description
13–8 NUM1[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
7	Reserved
6 CNF0	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
5–0 NUM0[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.

Figure 52-37 presents the XTRIG_CONF2 register; Table 52-37 provides its field descriptions.

0x83FD_4074 (XTRIG_CONF2) Access: User Read/Write
Wait State: 0

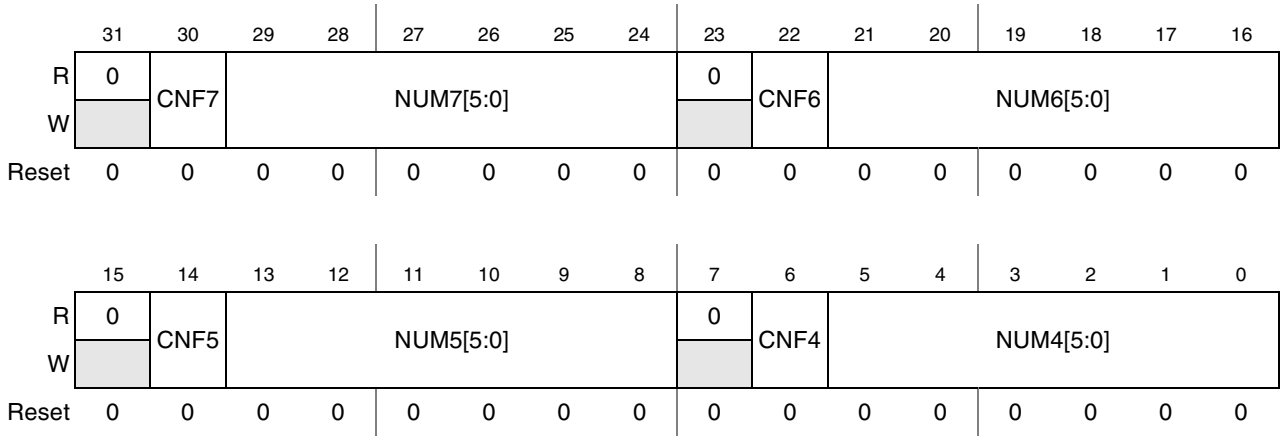


Figure 52-37. Cross-Trigger Events Configuration Register (2) (XTRIG_CONF2)

Table 52-37. XTRIG_CONF2 Field Descriptions

Field	Description
31	Reserved
30 CNF7	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
29–24 NUM7[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
23	Reserved

Table 52-37. XTRIG_CONF2 Field Descriptions (continued)

Field	Description
22 CNF6	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
21–16 NUM6[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
15	Reserved
14 CNF5	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution 0 channel 1 DMA request
13–8 NUM5[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.
7	Reserved
6 CNF4	Configuration of the SDMA event line number *i* that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution. 0 channel 1 DMA request
5–0 NUM4[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number *i*.

52.12.3.26 OnCE Trace Buffer Register (OTB)

Figure 52-38 shows the register; Table 52-38 provides its field descriptions.

0x83FD_40x078 (OTB) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	TBF	TADDR[13:2]											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TADDR[1:0]		CHFADDR[13:0]													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-38. OnCE Trace Buffer (OTB) Register

Table 52-38. OTB Field Descriptions

Field	Description
31–29	Reserved
28 TBF	The Trace Buffer Flag is set when the buffer contains the addresses of a valid change of flow. The contents of the buffer should be ignored otherwise. 0 Invalid information 1 Valid information
27–14 TADDR	The target address is the address taken after the execution of the change of flow instruction.
13–0 CHFADDR	The change of flow address is the address where the change of flow is taken when executing a change of flow instruction.

52.12.3.27 Profile Counter Registers (PRF_CNT_x)

Figure 52-40 shows the register; Table 52-40 provides its field descriptions.

0x83FD_40x07C (PRF_CNT_1)
to
0x83FD_40x090 (PRF_CNT_6)

Access: User Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	COUNTER_CONFIG									OFL	COUNTER					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNTER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-39. Profile Counter Register (PRF_CNT_x)

Table 52-39. PRF_CNT_x field description

Field	Description
31–23	These config bit are used to configure the counter to count the duration for which 000000000: SDMA core is in sleep state. 000001000: Burst DMA is in sleep state. 000001001: Burst DMA2 is in sleep state. 000001010: Reserved. 0001xxxxx: A particular channel is active. 5 x bit defines the channel number. 0010xxxxx: A particular channel is runnable but it is pending for arbitration. the 5 x bit defines the channel number. 0011xxxxx: SDMA core is in FUBUS state while running a particular channel. the 5 x bit defines the channel number. 011xxxxxx: SDMA request is active. the 6 x bit defines the DMA requests. 1000xxxxx: Burst DMA unit is in sleep state for a particular channel. the 5 x bit defines the channel. 1001xxxxx: Burst DMA2 unit is in sleep state for a particular channel. the 5 x bit defines the channel. 1010xxxxx: Reserved. 11xxxxxxxx: Free running counter. Only PRF_CNT_6 register can be programmed in this mode and 29th is the enable bit for it.
22 OFL	Overflow flag is status to show the profile counter has overflowed. It is reset by re-enabling the profile counter by setting the enable bit in PRF_CGF register. 1: profile counter has overflowed. 0: profile counter has not overflowed.
21–0 COUNTER	This 22 bit value shows the count for the particular profile mode selected by profile config [31-23] bits.

52.12.3.28 Profile Config/Status Register (PRF_CGF)

Figure 52-40 shows the register; Table 52-40 provides its field descriptions.

0x83FD_40x094 (PRF_CFG) Access: User Read-Write
wait: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	ISR	OFL6	OFL5	OFL4	OFL3	OFL2	OFL1	INT_ EN_6	INT_ EN_5	INT_ EN_4	INT_ EN_3	INT_ EN_2	INT_ EN_1	EN
W			w1c													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-40. Profile Config/Status Register (PRF_CGF)

Table 52-40. PRF_CGF Field Description

Field	Description
31–14	Reserved.
13 ISR	If any of the profile counter has overflowed, This bit is set. w1c clears this bit. This bit is set again only when all the overflow bit of counter is cleared once. 1: when the overflow occurs and if the INT_EN bit is set then interrupt is sent to the AP side and the interrupt gets cleared when w1c occurs. 0: no counter overflow has occurred.
12 OFL6	THe bit represent if the profile counter 6 has overflowed. 1: profile counter 6 has overflowed. 0: profile counter 6 has not overflowed.
11 OFL5	The bit represent if the profile counter 5 has overflowed. 1: profile counter 5 has overflowed. 0: profile counter 5 has not overflowed.
10 OFL4	The bit represent if the profile counter 4 has overflowed. 1: profile counter 4 has overflowed. 0: profile counter 4 has not overflowed.
9 OFL3	The bit represent if the profile counter 3 has overflowed. 1: profile counter 3 has overflowed. 0: profile counter 3 has not overflowed.
8 OFL2	The bit represent if the profile counter 2 has overflowed. 1: profile counter 2 has overflowed. 0: profile counter 2 has not overflowed.
7 OFL1	The bit represent if the profile counter 1 has overflowed. 1: profile counter 1 has overflowed. 0: profile counter 1 has not overflowed.
6 INT_EN_6	The bit represent if the interrupt is enabled for profile counter 6. 1: the interrupt is enabled for profile counter 6. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 6.
5 INT_EN_5	The bit represent if the interrupt is enabled for profile counter 5. 1: the interrupt is enabled for profile counter 5. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 5.
4 INT_EN_4	The bit represent if the interrupt is enabled for profile counter 4. 1: the interrupt is enabled for profile counter 4. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 4.
3 INT_EN_3	The bit represent if the interrupt is enabled for profile counter 3. 1: the interrupt is enabled for profile counter 3. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 3.
2 INT_EN_2	The bit represent if the interrupt is enabled for profile counter 2. 1: the interrupt is enabled for profile counter 2. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 2.

Table 52-40. PRF_CGF Field Description

Field	Description
1 INT_EN_1	The bit represent if the interrupt is enabled for profile counter 1. 1: the interrupt is enabled for profile counter 1. If the profile counter overflows and this bit is set. the interrupt will be sent to AP side. 0: the interrupt is disabled for the profile counter 1.
0 EN	This bit enables the profile counters. When the bit is set all the counter and their overflow bit are reset and the counter are enabled for different profile mode. All the counters are reset at the rising edge of this bit and all the counters are frozen at the falling edge. 1: Profile counters are enabled. 0: Profile counters is disabled.

52.12.3.29 Channel Priority Registers (CHNPRIn)

Figure 52-41 presents the register; Table 52-41 provides its field descriptions.

0x83FD_4100+n*4 (CHNPRIn[1]) Access: User Read/Write
Wait State: 0

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	CHNPRIn[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-41. Channel Priority Registers (CHNPRIn)

1 for n = 1 to 31

Table 52-41. CHNPRIn Field Descriptions

Field	Description
31–3	Reserved
2–0 CHNPRIn	This contains the priority of channel number *n*. Useful values are between 1 and 7; 0 is reserved by the SDMA hardware to determine when there is no pending channel. Reset value is 0, which prevents the channels from starting.

52.12.3.30 Channel Enable RAM (CHNENBLn)

Figure 52-42 presents the register; Table 52-42 provides its field descriptions.

0x83FD_4200+n*4 (CHNENBLn[1]) Access: User Read/Write
Wait State: 1

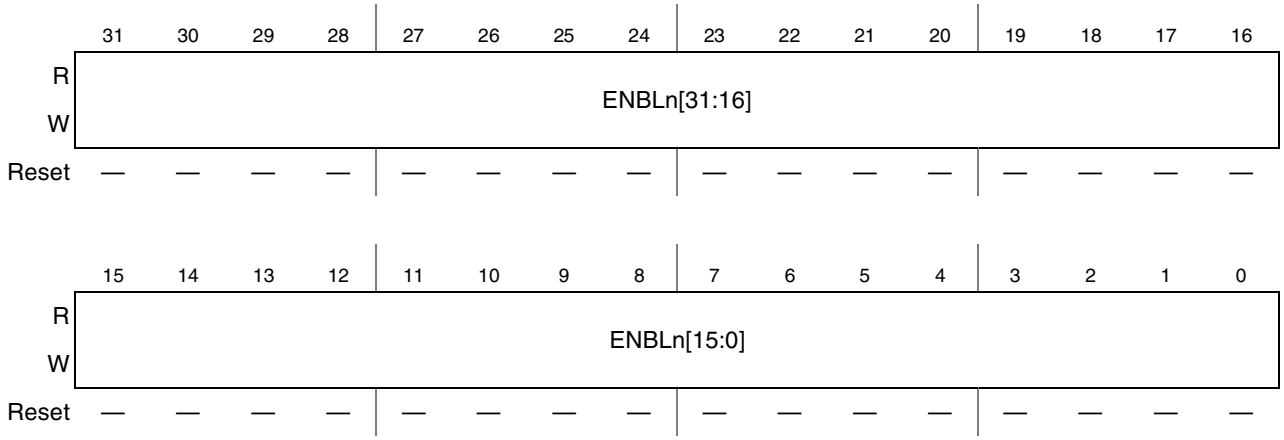


Figure 52-42. Channel Enable RAM (CHNENBLn) Register

[1] for n = 0 to 47

Table 52-42. CHNENBLn Field Descriptions

Field	Description
31–0 ENBLn	This 32-bit value selects the channels that are triggered by the DMA request number *n*. If ENBLn[i] is set to 1, bit EP[i] will be set when the DMA request *n* is received. These 48 32-bit registers are physically located in a RAM, with no known reset value. It is thus essential for the AP to program them before any DMA request is triggered to the SDMA, otherwise an unpredictable combination of channels may be started.

52.13 SDMA Programming Model

The following section describes the programming model for the SDMA RISC engine, including its processor, memory, and internal control registers.

All addresses are related to the internal SDMA memory map, which is completely different from the AP memory maps. The AP processor has no access to any hardware resource described, except when those resources are described in Section 52.12, AP Memory Map and Control Register Definitions."

52.13.1 State and Registers Per Channel

The SDMA can be seen as a set of 32 identical devices that are able to perform one data transfer channel each. Only one channel can work at a time, but every channel state is available at any time. This chapter lists the components of every channel state.

52.13.2 General Purpose Registers

Each channel has eight general purpose registers of 32 bits for use by scripts. General register 0 has a dedicated function for the `loop` instruction, but otherwise can be used for any purpose.

52.13.3 Functional Unit State

Each channel context has some state that is part of the functional units. The specific allocation of this state is part of the functional unit definition that is described in Section 52.18.1, Burst DMA Unit," Section 52.18.2, Burst DMA2 Unit." This state must be saved/restored on context switches.

52.13.3.1 Program Counter Register (PC)

The PC is 14 bits. Since instructions are 16 bits in width and all memory in the SDMA is 32 bits in width, the low order bit of the PC selects which half of the 32-bit word contains the current instruction. A low order bit of zero selects the most significant half of the word.[1]

52.13.3.2 Flags

Each channel has the following four flags:

- The T bit reflects the status of some arithmetic and test instructions. It is set when the result of an addition or a subtraction is zero and cleared otherwise. It is also the copy of the tested bits. Finally, it can also be set when the loop counter (GReg0) reaches zero. When the last instruction of the hardware loop is an operation that can modify the T flag, its effect on T is discarded and replaced by the GReg0 status.
- Two additional bits, SF and DF, are used to indicate error conditions resulting from loading data sources and storing to destinations, respectively. Access errors set these bits, and successful transactions clear them. They can also be cleared by specific instructions (`CLRF` and `loop`). The source fault (SF) is updated by the loads LD and LDF; the destination fault (DF) is updated by the stores ST and STF.

  Access errors are caused by several conditions including writing to the ROM, writing to a read-only memory mapped register, accessing an unmapped address, or any transfer error received by a peripheral when it is accessed.

  The SF and DF flags have a major impact on the behavior of the hardware loop: If SF or DF is set when starting a hardware loop and it is not masked by the `loop` instruction, the loop body will not be executed. Inside the loop body, if a load or store sets the corresponding SF or DF flag, the loop exits immediately. Testing the status of the T flag at the end of the loop (as well as testing both SF and DF) tells if the loop exited abnormally as any anticipated exit prevents GReg0 from reaching the zero value and thus setting the T flag. This is also valid if the fault occurs at the last instruction of the last loop.
- The last flag is the loop mode flag, LM, which is composed of two bits. The most significant bit indicates when the processor is currently operating in loop mode. It is set by the `loop` instruction and is cleared after execution of the last instruction of the last loop. The least significant bit is set

1. For example, big-Endian.

when the program counter points to the last instruction of a loop on the last path. It is used for a channel that is restored with this configuration to know that the next program counter is EPC. As with the dynamic context switch Greg0, which indicates when the program must get out of the loop, it can be restored only on the last instruction of the loop. This, however, is too late to fetch the next instruction after the loop.

52.13.3.3 Return Program Counter (RPC)

The RPC is 14 bits. It is set by the jump to the subroutine instructions and used by the return from the subroutine instructions. Instructions are available to transfer its contents to and from a general register.

52.13.3.4 Loop Mode Start Program Counter (SPC)

The SPC is 14 bits. It is set by the `loop` instruction to the location immediately following it.

52.13.3.5 Loop Mode End Program Counter (EPC)

The EPC is 14 bits. It is set by the `loop` instruction to the location of the next instruction after the loop.

52.13.4 Context Switching

Each channel has a separate context consisting of the eight general purpose registers and additional registers representing the state of the functional units. The active registers and functional units contain the context of the active channel. The context of inactive channels is stored in SDMA RAM, which is part of the SDMA address space.

In a function of the selected context switching mode (Section 52.4.4, Context Switching”), modified registers by the program can be saved in the channel RAM space while the program is going on. In every cycle, a write access to the RAM is possible.

On a `done` or `yield(ge)` instruction, SDMA goes into “real” context switching. In one of the dynamic modes, modified registers not previously saved, as well as the PC-Loop registers, are stored into the context area of the channel that will be closed. The new PC-Loop registers are loaded from the context area of the new channel. All other registers are restored while the program is executed, giving priority to registers used by the decoded instruction. Therefore, in the best case, only the PC and Loop registers should be saved and restored during this context-switching phase, which only requires five SDMA cycles.

In static mode, the context switch stores all registers in the old channel RAM space, and restores all registers from the new channel RAM space. It requires 26 SDMA cycles.

The address of the context memory for channel *i* is *CONTEXT_BASE* + 24**i* or *CONTEXT_BASE* + *32***i* where CONTEXT_BASE equals 0x0800. Table 52-43 presents the layout of a channel context in memory:

Table 52-43. Layout of a Channel Context in Memory for SDMA

OFFSET	31	30	29–16	15	14	13–0
0	SF	-	RPC	T	-	PC
1	LM		EPC	DF	-	SPC

Table 52-43. Layout of a Channel Context in Memory for SDMA (continued)

2	GR0
3	GR1
4	GR2
5	GR3
6	GR4
7	GR5
8	GR6
9	GR7
10	MDA (burst DMA)
11	MSA (burst DMA)
12	MS (burst DMA)
13	MD (burst DMA)
14	Reserved.
15	Reserved.
16	Reserved.
17	Reserved.
18	Reserved.
19	Reserved.
20	DSA[1] (Burst DMA2)
21	DSA[1] (Burst DMA2)
22	DS[1] (Burst DMA2)
23	DD[1] (Burst DMA2)
24	Scratch RAM (optional)
25	Scratch RAM (optional)
26	Scratch RAM (optional)
27	Scratch RAM (optional)
28	Scratch RAM (optional)
29	Scratch RAM (optional)
30	Scratch RAM (optional)
31	Scratch RAM (optional)

52.13.5 Address Space

The SDMA has four internal buses, as follows:

- The Instruction bus reads instructions from the memory. Its address map is described in Section 52.13.5.1, Instruction Memory Map."
- The Data bus (DMBUS) accesses the same memories as those visible on the Instruction bus, some memory-mapped registers (scheduler status and OnCE registers), and up to 14 peripherals. Its address map is described in Section 52.13.5.2, Data Memory Map."
- The Functional Units bus (FUBUS) accesses the Burst DMA2, burst DMA. The addressing mechanism is further detailed in Section 52.18, Functional Units Programming Model."
- The Context Switch bus reads/writes registers into context-switch RAM space. It is a 64-bit bus dedicated for accessing this RAM space for updating the context of the running channel. While the program is going on, this bus has the lowest priority compared to the Instruction and Data buses, except for restoring a register needed for the decoded instruction to be executed. On the save part of a context switch (when the PCU is in its slave state), this is the only one used. On the restore part, the Instruction bus has the priority to read the next instruction at the restored PC and otherwise the Context Switch bus is used. It is not possible to control the actual data transfers that occur on this bus.

52.13.5.1 Instruction Memory Map

The instruction memory map is based on a 14-bit address bus and a 16-bit data (instruction) bus. Each address corresponds to a 16-bit data location. Instructions are fetched from either program ROM or program RAM. An SDMA script is able to change the contents of the program RAM, which is also visible from the data bus.

The first two instruction locations (at 0 and 1) are special. Location 0 is where the PC is set on reset. Location 1 is where the PC is set upon the execution of an illegal instruction. It is expected that both of these locations will contain a `jmp` to handle routines.

Table 52-44. SDMA Instruction Memory Space

Device	SDMA Address (Hex)	Base Address Label	Module Name	WS	Description
ROM	0x0000 ↓ 0x07FF	SDMA_IBUS_ROM_ADDR	—	0	4 KB internal ROM with boot code and standard routines.
RAM	0x1000 ↓ 0x1FFF	SDMA_IBUS_RAM_ADDR	—	0	8 KB internal RAM with channels context and user data/routines.

52.13.5.2 Data Memory Map

All of the data accessible to SDMA scripts make up the data memory space of the SDMA. This address space has several components:

- ROM (also visible on the Instruction bus)

- RAM (also visible on the Instruction bus)
- Shared Peripherals Registers
- SDMA Internal Registers (scheduler, OnCE, and registers that are also accessible by the AP)

SDMA scripts can read and write to the context RAM, data RAM, shared peripheral registers, and internal registers.

The address range is 16 bits and the data width is 32 bits. Each address corresponds to a 32-bit data word. When accessing peripheral registers (USB and so on), the data width may be different. The exact address map for the peripherals depends on the project (as presented in each respective chapter).

The SDMA can perform only 32-bit access to shared peripheral registers. For this device, shared peripherals registers are aliased as if byte addressed. This is a consequence of connections through the SPBA shared peripheral bus outside of the SDMA. The result is that although address space 4 Kwords (16KB) is allocated for each peripheral, only the first 4 KB of the peripheral's register space can accessed. For example, the shared peripheral register at address 0x3000 is mapped also to addresses 0x3001, 0x3002, 0x3003. A read or write access to any of any of these 4 addresses will respond as if the access was to address 0x3000.

Data access is performed with *ld* and *st* instructions that take the address from a general purpose register in the core (GReg*n*). The mapping between the general purpose register contents and the address bus is given in Table 52-45:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sz	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
address															

Table 52-45. GReg*n* to DMBUS Address Mapping

Grayed bits are simply discarded but they must be cleared to ensure forward-script compatibility.

- sz (bit 31) indicates the peripheral data width: 0 is used for a 32-bit peripheral and 1 is used for a 16-bit peripheral.
- address (bits 15 down to 0) is the address of the accessed resource (internal memory, internal register, or shared peripheral).

Table 52-46. SDMA Data Memory Space

Device	SDMA Address (Hex)	Size	Description
ROM	0x0000 → 0x03FF	4 KB	4 KB internal ROM with boot code and standard routines
Reserved	0x0400 → 0x07FF	4 KB	4 KB Reserved
RAM	0x0800 → 0x0FFF	8 KB	8 KB internal RAM with channels contexts and user data/routines
per1	0x1000 → 0x1FFF	16 KB	*peripheral 1* memory space (4 KB peripheral's address space)
per2	0x2000 → 0x2FFF	16 KB	*peripheral 2* memory space (4 KB peripheral's address space)
per3	0x3000 → 0x3FFF	16 KB	*peripheral 3* memory space (4 KB peripheral's address space)

Table 52-46. SDMA Data Memory Space (continued)

Device	SDMA Address (Hex)	Size	Description
per4	0x4000 → 0x4FFF	16 KB	*peripheral 4* memory space (4 KB peripheral's address space)
per5	0x5000 → 0x5FFF	16 KB	*peripheral 5* memory space (4 KB peripheral's address space)
per6	0x6000 → 0x6FFF	16 KB	*peripheral 6* memory space (4 KB peripheral's address space)
Registers	0x7000 → 0x7FFF	16 KB	Memory mapped registers
per7	0x8000 → 0x8FFF	16 KB	*peripheral 7* memory space (4 KB peripheral's address space)
per8	0x9000 → 0x9FFF	16 KB	*peripheral 8* memory space (4 KB peripheral's address space)
per9	0xA000 → 0xAFFF	16 KB	*peripheral 9* memory space (4 KB peripheral's address space)
per10	0xB000 → 0xBFFF	16 KB	*peripheral 10* memory space (4 KB peripheral's address space)
per11	0xC000 → 0xCFFF	16 KB	*peripheral 11* memory space (4 KB peripheral's address space)
per12	0xD000 → 0xDFFF	16 KB	*peripheral 12* memory space (4 KB peripheral's address space)
per13	0xE000 → 0xEFFF	16 KB	*peripheral 13* memory space (4 KB peripheral's address space)
per14	0xF000 → 0xFFFF	16 KB	*peripheral 14* memory space (4 KB peripheral's address space)

52.14 SDMA Internal (Core) Memory Map and Internal Register Definitions

The actual SDMA memory mapped registers are summarized in the following sections; for peripherals' memory maps, refer to the respective chapters.

52.14.1 SDMA Internal (Core) Registers Memory Map

Table 52-47. SDMA Internal Registers Memory Map

Offset	Register	Access	Reset Value	Section/Page
General Registers				
0x7000 (MC0PTR)	AP (MCU) Channel 0 Pointer	R	0x0000_0000	52.14.3.1/52-89
0x7002 (CCPTR)	Current Channel Pointer	R	0x0000_0000	52.14.3.2/52-89
0x7003 (CCR)	Current Channel Register	R	0x0000_0000	52.14.3.3/52-90
0x7004 (NCR)	Highest Pending Channel Register	R	0x0000_0000	52.14.3.4/52-91
0x7005 (EVENTS)	External DMA Requests Mirror	R	0x0000_0000	52.14.3.5/52-91
0x7006 (CCPRI)	Current Channel Priority	R	0x0000_0000	52.14.3.6/52-93
0x7007 (NCPRI)	Next Channel Priority	R	0x0000_0000	52.14.3.7/52-93
0x7009 (ECOUNT)	OnCE Event Cell Counter	R/W	0x0000_0000	52.14.3.8/52-94
0x700A (ECTL)	OnCE Event Cell Control Register	R/W	0x0000_0000	52.14.3.9/52-95
0x700B (EAA)	OnCE Event Address Register A	R/W	0x0000_0000	52.14.3.10/52-97

Table 52-47. SDMA Internal Registers Memory Map (continued)

Offset	Register	Access	Reset Value	Section/Page
0x700C (EAB)	OnCE Event Cell Address Register B	R/W	0x0000_0000	52.14.3.11/52-97
0x700D (EAM)	OnCE Event Cell Address Mask	R/W	0x0000_0000	52.14.3.12/52-98
0x700E (ED)	OnCE Event Cell Data Register	R/W	0x0000_0000	52.14.3.13/52-99
0x700F (EDM)	OnCE Event Cell Data Mask	R/W	0x0000_0000	52.14.3.14/52-99
0x7018 (RTB)	OnCE Real-Time Buffer	R/W	0x0000_0000	52.14.3.15/52-100
0x7019 (TB)	OnCE Trace Buffer	R	0x0000_0000	52.14.3.16/52-101
0x701A (OSTAT)	OnCE Status	R	0x0000_0000	52.14.3.17/52-102
0x701C (MCHN0ADDR)	Channel 0 Boot Address	R	0x0000_0000	52.14.3.18/52-104
0x701D (MODE)	Mode Status Register	R	0x0000_0000	52.14.3.19/52-105
0x701E (LOCK)	Lock Status Register	R	0x0000_0000	52.14.3.20/52-106
0x701F (EVENTS2)	External DMA Requests Mirror #2	R	0x0000_0000	52.14.3.21/52-106
0x7020 (HE)	AP Enable Register	R	0x0000_0000	52.14.3.22/52-107
0x7022 (PRIV)	Current Channel BP Privilege Register	R	0x0000_0000	52.14.3.23/52-108
0x7023 (PRF_CNT)	Profile Free Running Register	R/W	0x0000_0000	52.14.3.24/52-109

52.14.2 Register Summary

The following definitions serve as a key for the SDMA internal register summary.

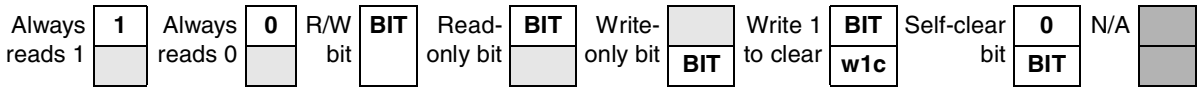


Figure 52-43. Key to Register Fields

Table 52-48 provides a key for register figures.

Table 52-48. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.

Table 52-48. Register Figure Conventions (continued)

Convention	Description
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Note: for n = 0 to 31

Table 52-49 presents a summary of the SDMA internal (core) registers.

Table 52-49. SDMA Internal Registers Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x7000 (MC0PTR)	R	MC0PTR[31:16]															
	W																
	R	MC0PTR[15:0]															
	W																
0x7001 RESERVED	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
0x7002 (CCPTR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	CCPTR[15:0]															
	W																
0x7003 (CCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	CCR[4:0]				
	W																
0x7004 (NCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	NCR[4:0]				
	W																
0x7005 (EVENTS)	R	EVENTS[31:16]															
	W																
	R	EVENTS[15:0]															
	W																
0x7006 (CCPRI)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	CCPRI[2:0]		
	W																

Table 52-49. SDMA Internal Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x7007 (NCPRI)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	NCPRI[2:0]		
	W																
0x7009 (ECOUNT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	ECOUNT[15:0]															
	W	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm
0x700A (ECTL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	EN	CNT	ECTC[1:0]		DTC[1:0]		ATC[1:0]		ABTC[1:0]		AATC[1:0]		ATS[1:0]	
	W																
0x700B (EAA)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	EAA[15:0]															
	W																
0x700C (EAB)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	EAB[15:0]															
	W																
0x700D (EAM)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	EAM[15:0]															
	W																
0x700E (ED)	R	ED[31:16]															
	W																
	R	ED[15:0]															
	W																
0x700F (EDM)	R	EDM[31:16]															
	W																
	R	EDM[15:0]															
	W																

Table 52-49. SDMA Internal Registers Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x7018 (RTB)	R	RTB[31:16]															
	W																
	R	RTB[15:0]															
	W																
0x7019 (TB)	R	0	0	0	TBF	TADDR[13:2]											
	W																
	R	TADDR[1:0]		CHFADDR[13:0]													
	W																
0x701A (OSTAT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0	0	0	0	ECDR[2:0]		
	W																
0x701C (MCHN0ADDR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	SMSZ	CHN0ADDR[13:0]													
	W																
0x701D (MODE)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	DSP Ctrl	0	0	AP-END
	W																
0x701E (LOCK)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LOCK
	W																

Table 52-49. SDMA Internal Registers Summary (continued)

Name		31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x701F (EVENTS2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	EVENTS[47:32]															
	W																
0x7020 (HE)	R	HE[31:16]															
	W																
	R	HE[15:0]															
	W																
0x7021 RESERVED	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
0x7022 (PRIV)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BP PRIV
	W																
0x7023 (PRF_CNT)	R	SEL		EN	RES						OFL	COUNTER					
	W																
	R	COUNTER															
	W																

52.14.3 SDMA Core Register Descriptions

The SDMA core has access to several memory mapped registers through its internal data bus. They are described in the following sections.

52.14.3.1 AP (MCU) Channel 0 Pointer (MC0PTR)

The following table shows the register; Table 52-50 provides its field descriptions.

0x7000 (MC0PTR) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MC0PTR[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MC0PTR[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-44. AP (MCU) Channel 0 Pointer (MC0PTR) Register

Table 52-50. MC0PTR Field Descriptions

Field	Description
31–0 MC0PTR	Contains the address—in the AP memory space—of the initial SDMA context and scripts that are loaded by the SDMA boot script running on channel 0.

52.14.3.2 Current Channel Pointer (CCPTR)

Figure 52-45 shows the register; Table 52-51 provides its field descriptions.

0x7002 (CCPTR) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CCPTR[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-45. Current Channel Pointer (CCPTR) Register

Table 52-51. CCPTR Field Descriptions

Field	Description
31–16	Reserved
15–0 CCPTR	Contains the start address of the context data for the current channel: Its value is *CONTEXT_BASE* + 24**CCR* or *CONTEXT_BASE* + 32**CCR* where CONTEXT_BASE = 0x0800. The value 24 or 32 is selected according to the programmed channel scratch RAM size in the register shown in Section 52.12.3.22, Channel 0 Boot Address (CHN0ADDR)."

52.14.3.3 Current Channel Register (CCR)

Figure 52-46 shows the register; Table 52-52 provides its field descriptions.

0x7003 (CCR) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	CCR[4:0]				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-46. Current Channel Register (CCR)

Table 52-52. CCR Field Descriptions

Field	Description
31–5	Reserved
4–0 CCR	Contains the number of the current running channel whose context is installed. In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel. The PST bits in the OSTAT register indicate when the SDMA is in sleep state.

52.14.3.4 Highest Pending Channel Register (NCR)

Figure 52-47 shows the register; Table 52-53 provides its field descriptions.

0x7004 (NCR) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	NCR[4:0]				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-47. Highest Pending Channel Register (NCR)

Table 52-53. NCR Field Descriptions

Field	Description
31–5	Reserved
4–0 NCR	Contains the number of the pending channel that the scheduler has selected to run next.

52.14.3.5 External DMA Requests Mirror (EVENTS)

Figure 52-48 shows the register; Table 52-54 provides its field descriptions.

0x7005 (EVENTS) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EVENTS[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-48. External DMA Requests Mirror (EVENTS)

Table 52-54. EVENTS Field Descriptions

Field	Description
31–0 EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs. This register displays EVENTS 0-31. The EVENTS2 register displays events 32-47.

NOTE

This register is very useful in the case of DMA requests that are active when a peripheral FIFO level is above the programmed watermark. The activation of the DMA request (rising edge) is detected by the SDMA logic and it can enable one or several channels. One of the channels accesses the peripheral and reads or writes a number of data that matches the watermark level (for example, if the watermark is four words, the channel reads or writes four words).

If the channel is effectively executed long after the DMA request was received, reading or writing the watermark number of data may not be sufficient to reset the DMA request (for example, if the FIFO watermark is four and at the channel execution it already contains nine pieces of data). This means no new rising edge may be detected by the SDMA, although there still remains transfers to perform. Therefore, if the channel were terminated at that time, it would not be restarted, causing potential overrun or underrun of the peripheral.

The proposed mechanism is for the channel to check this register after it has performed the "watermark" number of accesses to the peripheral. If the bit for the DMA request that triggers this channel is set, it means there is still another watermark number of data to transfer. This goes on until the bit is cleared. The same script can be used for multiple channels that require this behavior. The script can determine its channel number from the CCR register and infer the corresponding DMA request bit to check. It needs a reference table that is coherent with the request-channel matrix that the AP programmed.

52.14.3.6 Current Channel Priority (CCPRI)

Figure 52-49 shows the register; Table 52-55 provides its field descriptions.

0x7006 (CCPRI) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	CCPRI[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-49. Current Channel Priority (CCPRI) Register

Table 52-55. CCPRI Field Descriptions

Field	Description
31–3	Reserved
2–0 CCPRI	Contains the 3-bit priority of the channel whose context is installed. It is 0 when no channel is running. 0 no running channel 1–7 current channel priority

52.14.3.7 Next Channel Priority (NCPRI)

Figure 52-50 shows the register; Table 52-56 provides its field descriptions.

0x7007 (NCPRI) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	NCPRI[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-50. Next Channel Priority (NCPRI) Register

Table 52-56. NCPRI Field Descriptions

Field	Description
31–3	Reserved
2–0 NCPRI	Contains the 3-bit priority of the channel the scheduler has selected to run next. It is 0 when no other channel is pending.

52.14.3.8 OnCE Event Cell Counter (ECOUNT)

Figure 52-51 shows the register; Table 52-57 provides its field descriptions.

0x7009 (ECOUNT) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECOUNT[15:0]															
W	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm	rwm
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-51. OnCE Event Cell Counter (ECOUNT) Register

Table 52-57. ECOUNT Field Descriptions

Field	Description
31–16	Reserved
15–0 ECOUNT	The event cell counter contains the number of times minus one that an event detection must occur before generating a debug request. • This register should be written before any attempt to use the event detection counter during an event detection process. • The counter is cleared on a JTAG reset.

52.14.3.9 OnCE Event Cell Control Register (ECTL)

Figure 52-52 shows the register; Table 52-58 provides its field descriptions.

0x700A (ECTL) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	EN	CNT	ECTC[1:0]		DTC[1:0]		ATC[1:0]		ABTC[1:0]		AATC[1:0]		ATS[1:0]	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-52. OnCE Event Cell Control Register (ECTL)

Table 52-58. ECTL Field Descriptions

Field	Description
31–14	Reserved
13 EN	Event Cell Enable. If the EN bit is set, the event cell is allowed to generate debug requests (the cell is awakened). If it is cleared, the event detection unit is disabled and no hardware breakpoint is generated, but matching conditions are still reflected on the emulation pin. 0 Cell is disabled. 1 Cell is enabled.
12 CNT	Event Counter Enable. The event counter enable bit determines if the cell counter is used during the event detection. In order to use the event counter during an event detection process, the event cell counter register should be loaded with a value equal to the number of times minus one that an event occurs before a debug request is sent. After every event detection, the counter is decreased. When the counter reaches the value 0, the event detection cell sends a debug request to the core. The event counter register should be written and the EN bit should be set before each new event detection process uses the event counter. 0 Counter is disabled. 1 Counter is enabled.
11–10 ECTC[1:0]	The event cell trigger condition bits select the combination of address and data matching conditions that generate the final address/data condition. During program execution, if this event cell trigger condition goes to 1, a debug request is sent to the SDMA. The EN bit must be set to enable the debug request generation. 00 address ONLY 01 data ONLY 10 address AND data 11 address OR data

Table 52-58. ECTL Field Descriptions (continued)

Field	Description
9–8 DTC[1:0]	The data trigger condition bits define when data is considered matching after comparison with the data register of the event detection unit. The operations are performed on unsigned values. 00 equal 01 not equal 10 greater than 11 less than
7–6 ATC[1:0]	The address trigger condition bits select how the two address conditions (addressA and addressB) are combined to define the global address matching condition. The supported combinations are described, as follows. 00 addressA ONLY 01 addrA AND addrB 10 addrA OR addrB 11 reserved
5–4 ABTC[1:0]	The Address B Trigger Condition (ABTC) controls the operations performed by address comparator B. All operations are performed on unsigned values. This comparator B outputs the addressB condition. 00 equal 01 not equal 10 greater than 11 less than
3–2 AATC[1:0]	The Address A Trigger Condition (AATC) controls the operations performed by address comparator A. All operations are performed on unsigned values. This comparator A outputs the addressA condition. 00 equal 01 not equal 10 greater than 11 less than
1–0 ATS[1:0]	The access type select bits define the memory access type required on the SDMA memory bus. 00 read ONLY 01 write ONLY 10 read or write 11 —

52.14.3.10 OnCE Event Address Register A (EAA)

Figure 52-53 shows the register; Table 52-59 provides its field descriptions.

0x700B (EAA) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EAA[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-53. OnCE Event Address Register A (EAA)

Table 52-59. EAA Field Descriptions

Field	Description
31–16	Reserved
15–0 EAA	Event Cell Address Register A computes an address A condition. It is cleared on a JTAG reset.

52.14.3.11 OnCE Event Cell Address Register B (EAB)

Figure 52-54 shows the register; Table 52-60 provides its field descriptions.

0x700C (EAB) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EAB[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-54. OnCE Event Cell Address Register B

Table 52-60. EAB Field Descriptions

Field	Description
31–16	Reserved
15–0 EAB	Event Cell Address Register B computes an address B condition. It is cleared on a JTAG reset.

52.14.3.12 OnCE Event Cell Address Mask (EAM)

Figure 52-55 shows the register; Table 52-61 provides its field descriptions.

0x700D (EAM) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EAM[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-55. OnCE Event Cell Address Mask (EAM)

Table 52-61. EAM Field Descriptions

Field	Description
31–16	Reserved
15–0 EAM	The Event Cell Address Mask contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before performing the address comparison. **Note:** There is a common address mask value for both address comparators. If bit *i* of this register is set, then bit *i* of the address value latched from the memory bus does not influence the result of the address comparison. The register is cleared on a JTAG reset.

52.14.3.13 OnCE Event Cell Data Register (ED)

Figure 52-56 shows the register; Table 52-62 provides its field descriptions.

0x700E (ED) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ED[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ED[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-56. OnCE Event Cell Data (ED) Register

Table 52-62. ED Field Descriptions

Field	Description
31–0 ED	The event cell data register contains a user defined data value. This data value is an input for the data comparator which generates the data condition. It is cleared on a JTAG reset.

52.14.3.14 OnCE Event Cell Data Mask (EDM)

Figure 52-57 shows the register; Table 52-63 provides its field descriptions.

0x700F (EDM) Access: User Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EDM[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EDM[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-57. OnCE Event Cell Data Mask (EDM) Register

Table 52-63. EDM Field Descriptions

Field	Description
31–0 EDM	The event cell data mask register contains the user-defined data mask value. • This mask is applied to the data value latched from the memory bus before performing the data comparison. • Setting bit *i* of the event cell data mask register means that bit *i* of the data value latched from the address bus does not influence the result of the data comparison. • The data mask is cleared on a JTAG reset.

52.14.3.15 OnCE Real-Time Buffer (RTB)

Figure 52-58 shows the register; Table 52-64 provides its field descriptions.

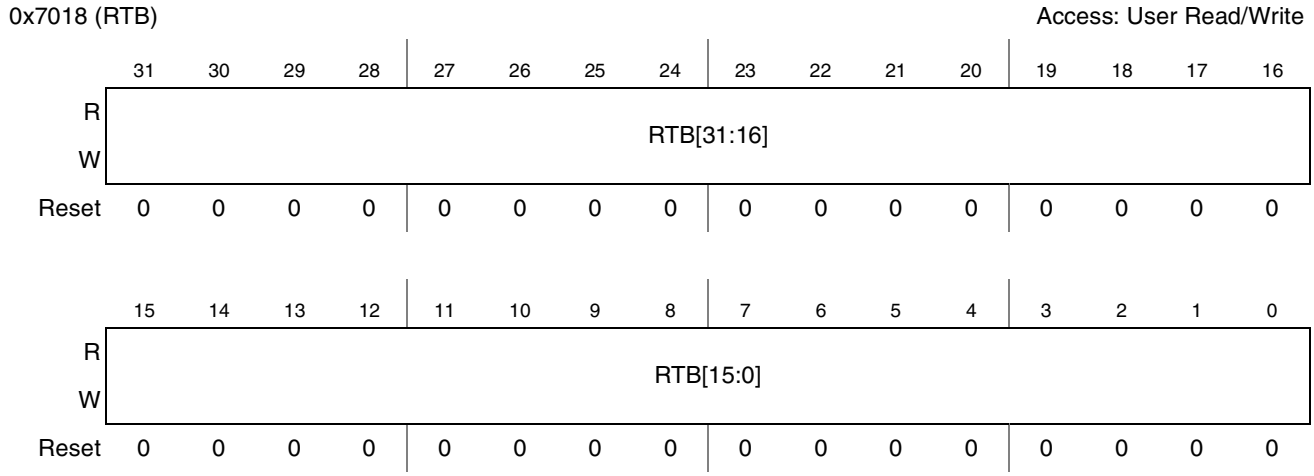


Figure 52-58. OnCE Real-Time Buffer (RTB) Register

Table 52-64. RTB Field Descriptions

Field	Description
31–0 RTB	The Real Time Buffer register stores and retrieves run time information without putting the SDMA in debug mode. Writing to that register triggers a pulse on a specific real-time debug pin whose connection depends on the chip implementation. The RTB value can be accessed by the OnCE under AP or JTAG control using the rbuffer command.

52.14.3.16 OnCE Trace Buffer (TB)

Figure 52-59 shows the register; Table 52-65 provides its field descriptions.

0x7019 (TB) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	TBF	TADDR[13:2]											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TADDR[1:0]		CHFADDR[13:0]													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-59. OnCE Trace Buffer (TB) Register

Table 52-65. TB Field Descriptions

Field	Description
31–29	Reserved
28 TBF	The Trace Buffer Flag is set when the buffer contains the addresses of a valid change of flow. The contents of the buffer should be ignored otherwise. 0 Invalid information 1 Valid information
27–14 TADDR	The target address is the address taken after the execution of the change of flow instruction.
13–0 CHFADDR	The change of flow address is the address where the change of flow is taken when executing a change of flow instruction.

52.14.3.17 OnCE Status (OSTAT)

Figure 52-60 shows the register; Table 52-66 provides its field descriptions.

0x701A (OSTAT) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0	0	0	0	ECDR[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-60. OnCE Status (OSTAT) Register

Table 52-66. OSTAT Field Descriptions

Field	Description
31–16	Reserved
15–12 PST[3:0]	The Processor Status bits reflect the state of the SDMA RISC engine. • The “Program” state is the usual instruction execution cycle. • The “Data” state is inserted when there are wait-states during a load or a store on the data bus (ld or st). • The “Change of Flow” state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel-switching instructions). • The “Change of Flow in Loop” state is used when an error causes a hardware loop exit. • The “Debug” state means the SDMA is in debug mode. • The “Functional Unit” state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In “Sleep” modes, no script is running (this is the RISC engine idle state). The “after Reset” is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation). • The “in Sleep” states are the same as above except they do not have any corresponding channel. They are used when entering debug mode after reset; the reason is that it is necessary to return to the “Sleep after Reset” state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Save 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change Flow Loop Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Restore
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an `rbuffer` command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.

Table 52-66. OSTAT Field Descriptions (continued)

Field	Description
7 MST	This flag is raised when the OnCE is controlled from the AP peripheral interface. 0 JTAG interface controls the OnCE. 1 AP peripheral interface controls the OnCE.
2–0 ECDR[2:0]	Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one EDR bit is set; the meaning of the encoding is as follows: EDR[0] 1 matched addressA condition EDR[1] 1 matched addressB condition EDR[2] 1 matched data condition The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addressA, addressB, and data conditions; the value of those fields is given by the EDR bits.

52.14.3.18 Channel 0 Boot Address (MCHN0ADDR)

Figure 52-61 shows the register; Table 52-67 provides its field descriptions.

0x701C (MCHN0ADDR) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SMSZ	CHN0ADDR[13:0]													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-61. Channel 0 Boot Address (MCHN0ADDR) Register

Table 52-67. MCHN0ADDR Field Descriptions

Field	Description
31–15	Reserved

Table 52-67. MCHN0ADDR Field Descriptions (continued)

Field	Description
14 SMSZ	The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context, which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism. 0 24 words per context 1 32 words per context
13–0 CHN0ADDR[13:0]	Contains the address of the channel 0 routine programmed by the AP; it is loaded into a general register at the very start of the boot and the SDMA jumps to the address it contains. By default, it points to the standard boot routine in ROM.

52.14.3.19 Mode Status Register (MODE)

Figure 52-62 shows the register; Table 52-68 provides its field descriptions.

0x701D (MODE) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	DSPctrl	0	0	APEND
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-62. Mode Status Register Register

Table 52-68. MODE Field Descriptions

Field	Description
31–4 & 2-1	Reserved
3 DSPCtrl	This bit specifies whether SDMA is controlled by single or two core (AP and BP core). This Bit is mapped to DSPDMA bit of Section 52.12.3.14, Configuration Register (CONFIG) in AP 0- Dual core to control SDMA 1- Single core to control SDMA
0 APEND	APEND indicates the Endian mode of the Burst DMA2 and Burst DMA interfaces. 0 - AP is in big Endian mode 1 - AP is in little Endian mode

52.14.3.20 Lock Status Register (LOCK)

Figure 52-63 shows the register; Table 52-69 provides its field descriptions.

0x701E (LOCK) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	LOCK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-63. Lock Status (LOCK) Register

Table 52-69. LOCK Field Descriptions

Field	Description
31–0	Reserved
0 LOCK	The LOCK bit reports the value of the LOCK bit in the SDMA_LOCK status register. SDMA software may use this value to determine if certain operations such as loading of new scripts is allowed. 0 - LOCK bit clear 1 - LOCK bit set

52.14.3.21 External DMA Requests Mirror #2 (EVENTS2)

Figure 52-64 shows the register; Table 52-70 provides its field descriptions.

0x701F (EVENTS2) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS[47:32]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-64. External DMA Requests #2 (EVENTS2) Register

Table 52-70. EVENTS2 Field Descriptions

Field	Description
31–16	Reserved
15-0 EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs. This register displays EVENTS 32-47. The separate EVENTS register displays events 0-31.

52.14.3.22 Host Enable Register (HE)

Figure 52-64 shows the register; Table 52-70 provides its field descriptions.

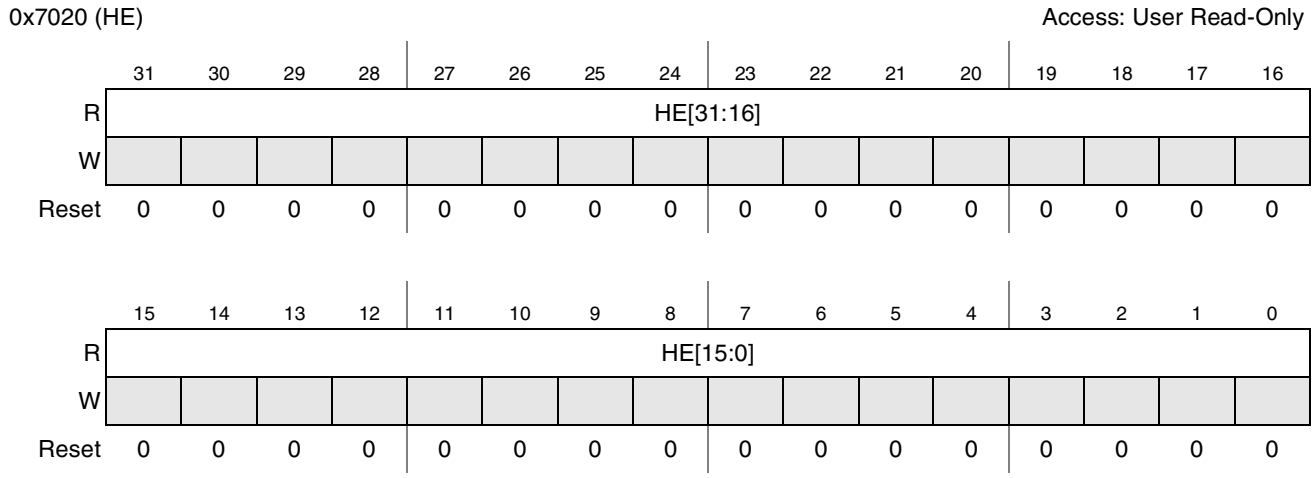


Figure 52-65. Host Enable (HE) Register

Table 52-71. HE Field Descriptions

Field	Description
31-0 HE	Reflects the status of the SDMA's host enable bits which are configured in the AP memory map by the HSTART or STOP_STAT registers.

52.14.3.23 Current Channel Privilege Register (PRIV)

Figure 52-64 shows the register; Table 52-70 provides its field descriptions.

0x7022 (PRIV) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	BP PRIV
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-66. Current Channel Privilege (PRIV) Register

Table 52-72. PRIV Field Descriptions

Field	Description
31–1	Reserved
0 BPPRIV	The BPPRIV bit indicates if the current channel has privilege to access BP memory. This bit is only updated when SDMA enters a context restore state based on the value of DE at the time. Thus, if DE is cleared while the channel is running, the privilege is retained until the next channel switch to allow any remaining DMA activity for that channel to complete. 0 - Current Channel does not have privilege to access BP memory 1 - Current Channel has privilege to access BP memory

52.14.3.24 Profile Free Running Counter Register (PRF_CNT)

Figure 52-67 shows the register; Table 52-73 provides its field descriptions.

0x7023 (PRF_CNT) Access: User Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEL		EN	RES						OFL	COUNTER					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNTER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 52-67. Profile Free Running Counter (PRF_CNT) Register

Table 52-73. PRF_CNT Field Descriptions

Field	Description
31–30 SEL	SEL bit shows if the profile counter is configured properly as free running counter. These can be configured by AP side by programming 31-30 bit of 0x83FD_40x090 (PRF_CNT_6). Also the profile counters should be enabled by using the 0 bit of 0x83FD_40x094 (PRF_CFG). 11: Counter is selected as free running counter. 10: Counter would not work as free running counter. 0x: Counter would not work as free running counter.:
29 EN	When the bit is set, the free running counter will start counting, provided the SEL bit shows 11 and also the profile counters should be enabled by using the 0 bit of 0x83FD_40x094 (PRF_CFG). 1: Start the free running counter. 0: free running counter is disabled.
28-23 RES	Reserved.
22 OFL	Profile free running counter has overflowed is represented by this bit. 1: Profile free running counter has overflowed. 2: Profile free running counter has not overflowed.
21-0 COUNTER	This shows the count value of the profile free running counter,

52.15 SDMA Peripheral Registers

Refer to the respective peripherals' chapters more information.

52.16 SDMA Initialization

The document *i.MX51 SDMA Scripts User Manual* describes the setup of the SDMA. This section provides a quick description of several initialization procedures.

NOTE

There may be differences with the actual implementation in the API.

52.16.1 Hardware Reset

After reset, the RAM that holds contexts, data, scripts, and the DMA request-channels matrix has unpredictable content. The core registers are all reset to 0, including the PC; the PCU state is *Sleep after Reset*. No channel can be activated because all of the priorities are also reset to 0.

52.16.2 Standard Boot Sequence

The following is the standard boot sequence:

1. Initialize the CONFIG register—detailed in Section 52.12.3.14, Configuration Register (CONFIG)”—to determine the AP DMA/core clock ratio (1 or 2)
2. Initialize the DMA request-channels matrix (see Section 52.12.3.30, Channel Enable RAM (CHNENBLn)”).
3. Program the channel control registers—Channel Event Override (EVTOVR), Channel BP Override (DSPOVR), Channel BP Override (HOSTOVR), and Channel Event Pending (EVTPEND)—according to the channel allocation.
4. Perform any necessary setup as required by the standard boot script in ROM (this is described in i.MX51 SDMA Scripts User Manual).
5. Trigger channel 0 with the Channel Start (HSTART) register, which starts the execution of the ROM script starting at address 0. This boot downloads channel scripts and contexts in RAM.

52.16.3 User-Defined Boot Sequence

The following is a user-defined boot sequence:

1. Initialize the Configuration Register (CONFIG), Channel Enable RAM (CHNENBLn), Channel Event Override (EVTOVR), Channel BP Override (DSPOVR), Channel AP Override (HOSTOVR), and Channel Event Pending (EVTPEND).
2. Use the OnCE (either via its JTAG interface or its AP control registers) to download any code in the SDMA RAM. Section 52.20.5.4, Accessing the Memory” describes how to write data to the RAM via the OnCE.
3. Use the OnCE instructions to make the PC default value point to the new boot script start address, or rely on the ROM startup script, which first jumps to the address in Channel 0 Boot Address (CHN0ADDR). (This register default address points to the standard boot script.)

52.16.4 Script Loading and Context Initialization

The execution of an SDMA script depends on both the instructions that make up the script and the data context upon which it operates. Both must be initialized before the script is allowed to execute. Each of the 32 channels has a separate data context, but may share scripts and locations in the data RAM.

The AP manages the space in program RAM and data RAM. It also manages the assignment of SDMA channels to the device drivers that need them. Channels are initialized by the AP via the channel 0 boot script. The boot channel downloads any required scripts with their data and the channels' initial contexts. Every context contains all the initial values of the registers, including the PC. Then the AP can enable any channel that becomes active and begins fetching and executing instructions from its script.

52.17 Instruction Description

The following sections introduce the instruction of the SDMA. Instruction set details are available in Section 52.21, Instruction Set."

52.17.1 Scheduling Instructions

The following are scheduling instructions:

- `done`—The instruction causes certain scheduling or interrupt bits to be set or cleared, which may cause a change in the schedule-ability of the running channel. Then the instruction causes the SDMA to evaluate the current scheduling priorities and to choose the highest priority ready channel. If this channel is not the current channel, a context switch will take place. If there are no runnable channels, the SDMA will enter the stopped mode. The `done` 5 has a special usage reserved for debug, as explained in Section 52.17.17, Debug Instructions."
- `yield`—These instructions are special cases of the `done` instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater than the current channel priority.
- `yieldge`—These instructions are special cases of the `done` instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater or equal to the current channel priority.
- `notify`—The `notify` instruction affects the scheduling bits, but does not cause rescheduling.

52.17.2 Conditional Branch Instructions

The conditional branch instructions of an 8-bit displacement, which is sign-extended and added to the current PC (which points to the next instruction) if the condition is satisfied. Otherwise, control passes to the next sequential instruction.

- `BF`—Branch if False. The branch is taken if the T bit in the processor status is zero (false).
- `BT`—Branch if True. The branch is taken if the T bit in the processor status is one (true).
- `BSF`—Branch if Source Fault. The branch is taken if the SF bit in the processor status is one.
- `BDF`—Branch if Destination Fault. The branch is taken if the DF bit in the processor status is one.

52.17.3 Unconditional Jump Instructions

There are two varieties of unconditional control transfers: an absolute transfer and a through-register transfer. Absolute transfers have a 14-bit address field that replaces the current PC.

- JMP—Jump. Causes the processor to jump to an absolute address encoded in the instruction itself.
- JSR—Jump to Subroutine. Causes the processor to jump to a subroutine, the address of which is encoded in the instruction itself.
- JMPR—Jump through Register. Causes the processor to jump to an absolute address contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.
- JSRR—Jump to Subroutine through Register. Causes the processor to jump to a subroutine, the address of which is contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.

52.17.4 Subroutine Return Instructions

The following are subroutine return instructions:

- RET—Return from Subroutine. The RET restores the contents of RPC to PC.
- LDRPC—Load from RPC to Register. THe LDRPC instruction is meant to be used when more than one level of subroutines are required. It stores the contents of RPC in any General register.

52.17.5 Loop Instruction

The following is a `loop` instruction:

LOOP—Enters Loop Mode. Before entering loop mode, the `loop` instruction can optionally clear the fault flags (SF and/or DF) based on a 2-bit field in the instruction. This feature is linked to the fact that setting SF or DF in loop mode will cause an immediate exit of the loop.

52.17.6 Miscellaneous Instructions

The following are miscellaneous instructions:

- CLRF—Clear Fault Flags. This instruction clears any combination of SF and DF.
- MOV r,s—This moves data from `GReg[s]` to `GReg[r]`.
- LDI r,immediate—This loads `GReg[r]` with a zero-extended immediate value.

52.17.7 Logic Instructions

The following are logic instructions:

- XOR r,s—This performs an `exclusive or` between `GReg[r]` and `GReg[s]`, and stores the result in `GReg[r]`.
- XORI r,immediate—This performs an `exclusive or` between `GReg[r]` and a zero-extended immediate value, and stores the result in `GReg[r]`.

- OR r,s—This performs an or between GReg[r] and GReg[s], and stores the result in GReg[r].
- ORI r,immediate—This performs an or between GReg[r] and a zero-extended immediate value and, stores the result in GReg[r].
- ANDN r,s—This performs an and between GReg[r] and the negated GReg[s], and stores the result in GReg[r].
- ANDNI r,immediate—This performs an and between GReg[r] and the negated zero-extended immediate value, and stores the result in GReg[r].
- AND r,s—This performs an and between GReg[r] and GReg[s], and stores the result in GReg[r].
- ANDI r,immediate—This performs an and between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].

52.17.8 Arithmetic Instructions

Arithmetic instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the result is zero, otherwise it is cleared.

- ADD r,s—This performs the addition of GReg[r] and GReg[s], and stores the result in GReg[r].
- ADDI r,immediate—This performs the addition of GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].
- SUB r,s—This performs the subtraction of GReg[s] from GReg[r], and stores the result in GReg[r].
- SUBI r,immediate—This performs the subtraction of a zero-extended immediate value from GReg[r], and stores the result in GReg[r].

52.17.9 Compare Instructions

Compare instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the comparison is true, otherwise it is cleared.

NOTE

Only one version of the immediate form is implemented. Non-equality comparisons to immediate values will require two instructions.

- CMPEQ r,s—This sets T when registers GReg[r] and GReg[s] are equal.
- CMPEQI r,immediate—This sets T when register GReg[r] and the zero-extended immediate value are equal.
- CMPLT r,s—This sets T when register GReg[r] is less than and not equal to GReg[s]. The comparison is signed.
- CMPHS r,s—This sets T when register GReg[r] is greater than or equal to GReg[s]. The comparison is signed.

52.17.10 Test Instructions

Test instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if any bit in the result is one, otherwise it is cleared.

- `TST r,s`—This performs an `and` between `GReg[r]` and `GReg[s]`, and sets T if the result is not zero.
- `TSTI r,immediate`—This performs an `and` between `GReg[r]` and a zero-extended immediate value, and sets `T` if the result is not zero.

52.17.11 Byte Permutation Instructions

These instructions shuffle the bytes in a register. For the purpose of describing these instructions, have the bytes in a register be numbered from the most significant as b_3, b_2, b_1, b_0.

- `RORB r`—The rotate right byte. The result is b_0, b_3, b_2, b_1.
- `REVB r`—The reverse bytes in word. The result is b_0, b_1, b_2, b_3.
- `REVBLO r`—The reverse, two low-order bytes. The result is b_3, b_2, b_0, b_1.

52.17.12 Bit Shift Instructions

The following are bit shift instructions:

- `ROR1 r`—The rotate right 1 bit. This instruction does a circular right shift of 1 bit.
- `LSR1 r`—The logical shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by a 0.
- `ASR1 r`—The arithmetic shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by itself.
- `LSL1 r`—The logical shift left 1 bit. This instruction shifts all bits to the left by 1. The low order bit is replaced by zero.

52.17.13 Bit Manipulation Instructions

- `BCLRI r,n`—The bit clear is immediate; clears bit number *i* in register *r*.
- `BSETI r,n`—The bit set is immediate; sets bit number *i* in register *r*.
- `BTSTI r,n`—The bit test is immediate; tests bit number *i* in register *r* (T becomes equal to the selected register bit).

52.17.14 SDMA Memory Access Instructions

All memory accesses are 32 bits. Any memory location that is implemented with less than 32 bits (for example, peripheral registers) causes unimplemented bits to be read as 0s. All memory accesses will cause either the SF or DF flags in the processor status to be set if they cause a fault. What constitutes a fault, especially when accessing peripheral registers, is a property of the memory location.

- `LD r,(b,d)`—The load instruction creates an address by adding the displacement field (d) to the contents of the base register (b). The SDMA location at the resulting address is read and placed in the destination register (r).
- `ST r,(b,d)`—The store instruction creates an address in the same manner as the load instruction. The register (r) is stored in the SDMA location at the resulting address.

52.17.15 Functional Unit Instructions

The functional unit instructions have an 8-bit field that is placed on the functional unit bus. Some of these bits are used to select which functional unit should be involved in the transfer. The remaining bits are decoded by the selected functional unit so their specific use depends on the functional unit. See Section 52.18, Functional Units Programming Model.”

There are two functional unit instructions, as follows:

- `LDF r,fub`—The 8-bit field is placed on the functional unit bus and a read is issued to the selected functional unit. As a result of this instruction, the SF may be set in the processor status.
- `STF r,fub`—The 8-bit field is placed on the functional unit bus and a write is issued to the selected functional unit. As a result of this instruction, the DF may be set in the processor status.

52.17.16 Illegal Instructions

All instruction encodings that are illegal cause the following actions:

- The current PC (which points to one beyond the offending instruction) is put in the EPC register.
- The loop mode bit is cleared.
- The PC is set to the value stored in the Illegal Instruction Trap Address (ILLINSTADDR) register (the default value is 0x0001).

ILLEGAL—Although any instruction other than those indicated in the SDMA specification will trigger the illegal instruction mechanism, the ILLEGAL instruction code is preferred as it will always be kept as *illegal* in the possible future versions of the SDMA core.

52.17.17 Debug Instructions

The following are debug instructions:

- `SOFTBKPT`—The software breakpoint instruction causes the core to stop and enter debug mode. The core can then be accessed and started by the OnCE debug module only.
- `done 5`—This instruction is used for debugging, as it copies the contents of the PCU registers and flags to the context memory. Information on this instruction is described in Section 52.20.5.2, Saving the Context.”
- `CpShReg`—This instruction copies the context memory into the PCU registers and flags. Modifying the corresponding memory location before executing this instruction enables you to have the channel continue from a new instruction address. This instruction is described in Section 52.20.5.3, Restoring the Context.”

52.18 Functional Units Programming Model

The functional unit instructions cause an 8-bit code, found in the low eight bits of the instruction, to be asserted on the functional unit control bus. Some of these bits are used to select one of several functional units. Functional units which can be selected include SDMA registers such as MSA and MSD which are not mapped in the SDMA memory map, and are accessible only through the functional unit bus. These Functional Unit Registers are listed in Table 52-74. In order to establish a programming convention, assume the selection bits are some number of the most significant bits of the 8-bit code. Furthermore, some number of the least significant bits is decoded by a given functional unit to establish the type of operation to perform.

Table 52-74. Functional Unit Registers

Functional Unit	Register	Register Name	Section/Page
Burst DMA Unit	MSA	Memory Source Address Register	52.18.1.1/52-116
	MDA	Memory Source Address Register	52.18.1.2/52-117
	MD	Memory Data Buffer Register	52.18.1.3/52-117 (Write) 52.18.1.5/52-119 (Read) 52.18.1.6/52-122
	MS	Memory State Register	52.18.1.4/52-118
Burst DMA2 Unit	DSA	BP Source Address Register	52.18.2.1/52-130
	DDA	BP Source Address Register	52.18.2.2/52-130
	DD	BP Data Buffer Register	52.18.2.3/52-131 (Write) 52.18.2.5/52-132 (Read) 52.18.2.6/52-135
	DS	BP State Register	52.18.2.4/52-131

More information regarding the functional units can be found in Section 52.6.2, Burst DMA2 Unit," and Section 52.6.1, Burst DMA Unit".

52.18.1 Burst DMA Unit

The DMA instructions control the DMA state machine and may cause a DMA cycle on the associated memory bus. There are four registers associated with the burst DMA unit, a Memory Source Address register (MSA), a Memory Destination Address register (MDA), a Memory Data buffer (MD), and a state register (MS).
The burst DMA has two different uses:

- A data transfer between External Memory Interface and SDMA general register
- A data transfer in copy mode where blocks of data are transferred from the source address to the destination address

52.18.1.1 Memory Source Address Register (MSA)

The source address register contains the pointer into EMI memory associated with the next read data transfer. It has byte granularity.

Reading the register with the `ldf` instruction has no side effects, and gives the address value in the EMI memory of the next data that is read by the SDMA during an `ldf MD` instruction.

Writing the source address register has two side effects: If the prefetch bit is set, a DMA read cycle (8-word read access) is issued with the new address. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MSA to guarantee all the data is effectively written to memory.

The MSA register has two modes of programming:

- Frozen—In frozen mode, the MSA register is not modified after DMA accesses.
- Incremented (default mode)—In incremental mode, MSA is incremented by the number of bytes transferred during read cycles.

52.18.1.2 Memory Destination Address Register (MDA)

The destination address register contains the pointer into EMI memory associated with the next write data transfer. It has byte granularity.

Reading the MDA register with the `ldf` instruction has no side effects. It gives the address value in the EMI memory where the next SDMA data (`stf r,MD` instruction) is stored when MD FIFO is flushed.

Writing the destination address register has one side effect. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MDA to guarantee all the data is effectively written to memory.

The MDA register has two modes of programming:

- Frozen—In frozen mode, the MDA register is not modified after DMA accesses.
- Incremented (default mode)—The MDA register is incremented by the number of bytes transferred during write cycles.

52.18.1.3 Memory Data Buffer Register (MD)

The data buffer register consists of a bank of 36 bytes that behave like FIFO. This FIFO stores the eight words received when a read burst is triggered by the DMA (DMA is in read mode). The MD register is in write mode after a writing in MDA or after an `stf MD` instruction. In that case, a burst write access is automatically triggered when there are more than eight words in MD. For bandwidth optimization, any transfers between DMA and the EMI controller are based on burst accesses.

An `ldf r,MD|SIZE` instruction that reads the data buffer may cause a DMA cycle, as follows:

- If there are less bytes in the FIFO than the size parameter of the instruction. For instance, if only two bytes are available in MD and a 4-byte read is requested, a burst read access is executed to complete the two bytes.
- If the prefetch bit is set, and after reading there is enough space in the FIFO to store a full burst, a burst read access is triggered.

An `stf r,MD|SIZE` instruction that writes to the data buffer may cause a DMA cycle if the number of written bytes in MD is higher than 32 (eight words) or if the flush bit is set.

When DMA is used for data transfer between SDMA and EMI (reading or writing), no immediate error is possible because the module manages a data misalignment issue; therefore, it is allowed to read/write a word to/from a half-word address. However, the addresses (source or destination) must belong to the EMI memory mapping. The only potential error, in this mode, would be the error sent back by the EMI controller when an access to a super-user page is detected. The whole transfer on the DMA associated bus will be considered successful when there are no errors seen on the bus during the transfer. In copy mode, an immediate error could be returned to SDMA as described in Section 52.18.1.10, Error Management."

52.18.1.4 State Register (MS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer. MS is also accessed to set-up the conditional yielding feature.

The initialization value of this register is 0 and it consists of the following:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	spriv	stype	0	0	dpriv	dtype
W																
R	0	0	0	0	y	d	e		0	0	n					
W																

Figure 52-68. MS Structure

Table 52-75. MS Field Descriptions

Field	Description
31–22	Reserved
21 spriv	The spriv value is ignored for this device. 0 = valid value 1 = Reserved
20 stype	Source Mode. Indicates if MSA has to be incremented (or not) during accesses. 0 Frozen—MSA is not modified. 1 Incremented—MSA is incremented by the number of transferred bytes during read access.
19–18	Reserved
17 dpriv	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved
16 dtype	Destination Mode. Indicates if MDA has to be incremented (or not) during accesses. 0 Frozen—MDA is not modified. 1 Incremented—MDA is incremented by the number of transferred bytes during write access.
15–12	Reserved

Table 52-75. MS Field Descriptions (continued)

Field	Description
11 y	Conditional Yielding selector. When selected, the `yield`/`yieldge` instructions will not switch channels if the Burst DMA is in Write Mode, and it has less than four bytes in its FIFO. This is aimed at reducing the number of inefficient FIFO flushes due to context switches. 0 Always yields 1 Yields conditionally (when there are less than four bytes in the FIFO in write mode)
10 d	Access Direction or DMA Mode. DMA is in write mode when data was written into MD by `stf MD` instructions, or if a previous DMA cycle on the external bus was a write access. Writing MDA or MSA changes the DMA mode to the respective value. DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by SDMA with an `ldf MD` instruction. Reading MDA or MSA does not change the DMA mode. 0 Read Mode 1 Write Mode
9–8 e	Error. Indicates if the previous access was acknowledged with a bus error. 00 No error was received. 01 *reserved* 10 Error mode 11 error read burst
7–6	Reserved
5–0 n	Number of bytes in the MD FIFO.

52.18.1.5 Burst DMA Write (stf)

When received from a `stf` instruction, the function code bits are interpreted as follows, depending on the addressed register:

Register	7	6	5	4	3	2	1	0
MSA	s		p	freeze	r			spriv
MDA								dpriv
MD			f	cpy			sz	
MS								

Figure 52-69. STF Code Bits

Table 52-76. STF Code Bit Field Descriptions

Field	Description
7–6 s	Functional Unit selector 00 for Burst DMA
5 p (MSA)	Prefetch Flag 0 No prefetch 1 Prefetch required from new MSA

Table 52-76. STF Code Bit Field Descriptions (continued)

Field	Description
5 f (MD)	Forced Flush Flag 0 Automatic flush 1 FIFO contents are flushed (including the new written data).
4 freeze (MSA/MDA)	Address Freeze Mode 0 Address is normally incremented. 1 Address is frozen.
4 cpy (MD)	Copy Mode selection 0 Write Mode 1 Copy Mode
3–2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1–0 sz (MD/MS)	Transfer Size 00 size 0 (no data stored in the FIFO) 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
0 spriv (MSA)	The spriv value is ignored for this device. 0 = valid value 1 = Reserved
0 dpriv (MDA)	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved

The possible write instructions are listed in Table 52-77 (unused bits should always be cleared).

Table 52-77. Burst DMA STF Instruction List

Binary	Assembly	Comments
00_0_0_00_00	stf r,MSA	Writes content of the SDMA general register (r) to the source address register. MSA is in incremented mode.
00_0_1_00_00	stf r,MSA|FR	Writes content of the SDMA general register (r) to the source address register. MSA is in frozen mode.
00_1_0_00_00	stf r,MSA|PF	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access. MSA is in incremented mode.
00_1_1_00_00	stf r,MSA|PF|FR	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access.
00_0_0_01_00	stf r,MDA	Writes content of the SDMA general register (r) to the destination address register. MDA is in incremented mode.
00_0_1_01_00	stf r,MDA|FR	Writes content of the SDMA general register (r) to the destination address register. MDA is in frozen mode.

Table 52-77. Burst DMA STF Instruction List (continued)

Binary	Assembly	Comments
00_1_0_10_00	stf r,MD|SZ0|FL	No data transfers between the SDMA and MD, but all valid written data of the MD is flushed to the memory. An acknowledge or error is sent back to the SDMA core on transfer completion.
00_0_0_10_01	stf r,MD|SZ8	8-bit (byte) transfer to write buffer MD
00_1_0_10_01	stf r,MD|SZ8|FL	8-bit (byte) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_10	stf r,MD|SZ16	16-bit (half-word) transfer to write buffer MD
00_1_0_10_10	stf r,MD|SZ16|FL	16-bit (half-word) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_11	stf r,MD|SZ32	32-bit (word) transfer to write buffer MD
00_1_0_10_11	stf r,MD|SZ32|FL	32-bit (word) transfer to write buffer MD and flush after transfer. All valid written data of MD is flushed to memory.
00_0_1_10_00	stf r,MD|CPY	No data transfer between SDMA and MD but starts a copy transfer whose length is given by the 4 LSB of *r* register. (Maximum burst length is eight words.)
00_0_0_11_11	stf r,MS	32-bit (word) transfer to status register MS
00_0_0_11_00	stf r,MS|SZ0	Clears the error flag (if set). Other MS bits are unchanged; this instruction is also known as `clref` MS.

NOTE

When a flush bit is set, the SDMA flushes the FIFO including the newly written data. An acknowledge is sent to the core before the flush completes (except if size 0 is used). The goal of this flush bit is to force a flush, but it is recommended to use it only when needed (for example, when finishing a row of pixels during 2D data transfers). Indeed, if this bit is omitted and if there are more than 32 bytes in the FIFO, a burst write access is automatically triggered.

Since all the `stf r,MD` instructions (including the copy mode) acknowledge the SDMA core before the store is effective (except if size 0 is used), it is recommended to perform an `ldf` from MS before terminating a channel in order to check the final error status. (The `ldf` from MS will stall the core until all the data was flushed out and the transfer status is known.)

After every `stf MD` instruction, the MDA is incremented by the number of bytes that are written in MD, except when it is programmed in frozen mode.

52.18.1.6 Burst DMA Read (ldf)

When received from an `ldf` instruction, the function code bits are interpreted as follows, depending on the addressed register:

Register	7	6	5	4	3	2	1	0
MSA	s				r			
MDA								
MD			p				sz	
MS								

Figure 52-70. LDF Code Bits

Table 52-78. LDF Code Bit Field Descriptions

Field	Description
7–6 s	Functional Unit selector 00 for Burst DMA
5 p (MD)	Prefetch Flag 0 no prefetch 1 automatic prefetch
3–2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1–0 sz (MD)	Transfer Size 00 reserved 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)

Table 52-79 lists the possible write instructions (unused bits should always be cleared).

Table 52-79. Burst DMA LDF Instruction List

Binary	Assembly	Comments
00_0_0_00_00	ldf r,MSA	Copies the source address register value into an SDMA general register. It gives the memory address of the next data that will be read with an `ldf` MD instruction.
00_0_0_01_00	ldf r,MDA	Copies the destination address register value into an SDMA general register. It gives the memory address where the next incoming data will be flushed.
00_0_0_10_01	ldf r,MD|SZ8	8-bit (byte) read
00_1_0_10_01	ldf r,MD|SZ8|PF	8-bit (byte) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_10	ldf r,MD|SZ16	16-bit (half-word) read

Table 52-79. Burst DMA LDF Instruction List (continued)

Binary	Assembly	Comments
00_1_0_10_10	ldf r,MD|SZ16|PF	16-bit (half-word) read. If after this reading, and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_11	ldf r,MD|SZ32	32-bit (word) read
00_1_0_10_11	ldf r,MD|SZ32|PF	32-bit (word) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_11_00	ldf r,MS	Copy the status register value into an SDMA general register.

NOTE

Read data is 0-extended before writing in the SDMA general registers. When reading the MD register, the DMA takes data from the FIFO if it is available. If part or whole data is not in the FIFO, an external burst read access is performed to provide the missing data. The SDMA is stalled as long as the required read data is not complete.

After every reading, MSA is incremented by the number of read bytes from MD FIFO, except when MSA is programmed in frozen mode.

52.18.1.7 Prefetch/Flush and Auto-Flush Management

The prefetch and auto-flush management enables the SDMA RISC machine to go on while a DMA access is performed. When the RISC core requires a prefetch (p = 1) to the Burst DMA, it will receive an immediate transfer acknowledge before the DMA has finished the external access. This enables the RISC core to do other things like accessing another DMA machine.

The basic principle in prefetch mode is for the DMA to anticipate data reads from the SDMA RISC engine by fetching external bursts of data as soon as there is enough space in the DMA FIFO to store it. If ever the RISC engine required data that is not available in the FIFO, the read acknowledge is delayed until the data is available, but it does not have to wait until the burst completes.

The auto-flush basic principle is similar: An automatic flush is triggered every time there are eight words to be written in the FIFO. If the FIFO is full and the RISC engine requires another write, it is stalled until the burst has started and enough space was freed in the FIFO to store that new data. This means the SDMA RISC engine does not have to wait for the completion of a burst to receive its acknowledge and continue its processing.

In particular, an auto-flush is executed when DMA is in write mode and if the following is true:

- If the FIFO is empty and the first write is to a word-aligned address of any size (ex: the 2 LSB of MDA[1:0]= 0x0), the auto-flush is triggered immediately after the write of the 32’nd byte.
- If the FIFO is empty, and if MDA is an odd byte address (1, 3, 5, 7,...) and an `stf MD|SZ8` is executed, the byte is flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.

- If the FIFO is empty, and if MDA is a half-word address (2, 6, 0xA,...) and an `stf MD|SZ16` is executed, the two bytes of the incoming data are flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if MDA is not a word-aligned address (ex 1, 2, 3, 5, 6, 7, 9,...), and an `stf MD|SZ32` is executed, the first 1 to 3 bytes will be flushed up to the next word aligned address. Afterwards, an auto-flush will be triggered each time the FIFO receives 32-bytes.

  Therefore, if an `stf MD|SZ32` is executed with MDA equal to 0x1 and with an empty MD FIFO, the bytes located at addresses 1, 2, and 3 are flushed, and the byte located at address 4 remains in MD FIFO. This solves the misalignment issue. Additionally, the next write instructions (`stf`) complete the FIFO until it contains eight words; then a burst write is executed by the DMA to empty the FIFO. Protocol on the external bus does not support bursts of different data types (byte, half-word, or word).

For example, consider the case where data is written using a byte access, `stf MD|SZ8`. The value of MDA during the very first byte write determines when the auto-flush will occur as follows:

- If MDA=0x0, the flush occurs following the write of byte 32
- If MDA=0x1, the flush occurs following the write of byte 1, byte 3 and byte 35.
- If MDA=0x2, the flush occurs following the write of byte 2 and byte 34.
- If MDA=0x3, the flush occurs following the write of byte 1 and byte 33.
- If MDA=0x4, the flush occurs following the write of byte 32

The flush command forces the DMA to flush all MD valid bytes to the EMI controller. An acknowledge is sent immediately to the SDMA, and any potential error is reported on a future access. It is thus essential to conclude a transfer with a last read from MS, which will stall the core until all data was flushed out and returned to the transfer status (acknowledge or error).

NOTE

During this kind of auto-flush—which occurs only at the beginning of a misaligned write transfer—no acknowledge is sent back to the SDMA, which is stalled until a flush is completed.

52.18.1.8 Data Alignment and Endianness

52.18.1.8.1 Burst DMA in Read Mode

For every read access to MD, the data returned to the SDMA core and the new FIFO state depends on the MSA status and the access size. The FIFO is considered as a stack of 36 bytes: Data is fetched externally on a 32-bit bus, but the valid bytes only are stored in the FIFO and left-aligned (for a transfer of consecutive words, it is only the first word that may be truncated). Table 52-80 shows the FIFO byte alignment strategy and the corresponding MSA, the returned data, and the new FIFO state for any access size of an internal read from MD.

Table 52-80. FIFO Read Configuration

Before read		Internal read access size	Read data	After read	
MSA[1:0]	FIFO state			MSA[1:0]	FIFO state
00	x0 x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 and so on...	sz8	00 00 00 x0	01	x1 x2 x3 y0 y1 y2 y3 z0
		sz16	00 00 x0 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz32	x0 x1 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
01	x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 and so on...	sz8	00 00 00 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz16	00 00 x1 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz32	x1 x2 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
10	x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 and so on...	sz8	00 00 00 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz16	00 00 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz32	x2 x3 y0 y1	10	y2 y3 z0 z1 z2 z3 t0 t1
11	x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 t2 and so on...	sz8	00 00 00 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz16	00 00 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
		sz32	x3 y0 y1 y2	11	y3 z0 z1 z2 z3 t0 t1 t2

52.18.1.8.2 Burst DMA in Write Mode

For every write access to the MD, the new FIFO state depends on the MDA status and the access size. The FIFO is considered as a stack of 36 bytes: Data is stored in the FIFO according to the internal access size and the former MDA value. Table 52-81 shows the FIFO byte alignment strategy corresponding to MDA, as well as the new FIFO state for any access size of an internal write to MD.

Table 52-81. FIFO Write Configuration

Before write		Internal write access size	Written data	After write	
MDA[1:0]	FIFO state			MDA[1:0]	FIFO state
00	tt uu vv ww ?? ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	01	tt uu vv ww x0 ?? ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	10	tt uu vv ww x0 x1 ?? ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	00	tt uu vv ww x0 x1 x2 x3 ?? ?? ?? ??
01	tt uu vv ww xx ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	10	tt uu vv ww xx x0 ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	11	tt uu vv ww xx x0 x1 ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	01	tt uu vv ww xx x0 x1 x2 x3 ?? ?? ??
10	tt uu vv ww xx yy ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	11	tt uu vv ww xx yy x0 ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	00	tt uu vv ww xx yy x0 x1 ?? ?? ?? ??
		sz32	x0 x1 x2 x3	10	tt uu vv ww xx yy x0 x1 x2 x3 ?? ??
11	tt uu vv ww xx yy zz ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	00	tt uu vv ww xx yy zz x0 ?? ?? ?? ??
		sz16	?? ?? x0 x1	01	tt uu vv ww xx yy zz x0 x1 ?? ?? ??
		sz32	x0 x1 x2 x3	11	tt uu vv ww xx yy zz x0 x1 x2 x3 ??

NOTE

If the FIFO mode changes from a write to a read mode, all remaining written bytes in MD are lost but no error is returned. Typically, this happens if an `ldf` MD is executed after `stf` MD instructions. Before a mode change, it is recommended to force the flush of a potential remaining byte by a `stf MD|SZ0|FL` instruction. In the same way, if a FIFO mode changes from a read to a write mode, all prefetched data present in the FIFO is lost and no error is returned.

52.18.1.8.3 Endianness

Big and Little Endian are supported by the Burst DMA, but data is always stored in MD in Big Endian. Byte manipulation is performed when data is exchanged with an Burst controller (for example, during read or write burst accesses).

52.18.1.9 Copy Mode

A mechanism is available to perform fast AP-to-AP transfers. Data does not flow through the SDMA core: It is kept in the DMA FIFO. This mechanism is selected when writing MD with a special option in the instruction code (copy flag). It is possible to transfer up to eight words in one SDMA instruction (this does not mean in one cycle). In this mode, every time an `stf MD|CPY` is executed, a read burst is executed and directly followed by a write burst transfer. Burst transfers are limited to eight words. The size of the transfer (in words)—given by the SDMA general register (4 LSB)—is also limited to eight. The following SDMA code shows how 100 bytes could be copied from the MSA address to the MDA address. This is sample code only.

Example 52-1. Burst DMA copy mode example

```
        ldi r0,@src
        stf r0,MSA                  // Source address setup
        ldi r1,@dst
        stf r1,MSA                  // Destination address setup
        ldi r0,0x64                 // data transfer counter
        ldi r1,0x8
MAIN_XFER:
        cmphs r0,r1                 // Is r0 >= 0x8
        bf LAST_XFER                // If not, jump to last transfer label
        stf r1,MD|CPY               // Copy 8 words from MSA to MDA address.
        subi r0,0x8                 // Decrement counter
        jmp MAIN_XFER               // return to main transfer loop
LAST_XFER:
        stf r0,MD|CPY
```

The main transfer loop is executed 12 times; then `r0` equals 4 and the last transfer loop is run.

In this mode, an acknowledge is transmitted to the core as soon as the read burst can start; thus, a first copy instruction returns an immediate acknowledge and subsequent copy instructions will be acknowledged as soon as the previous copy has finished.

52.18.1.10 Error Management

Another point to consider is the management of errors. Because the DMA immediately sends an acknowledge to the RISC core (except for the `stf MS|SZ0|FLS` instruction), it assumes no error will occur. If an error occurs, it is flagged (transfer error acknowledge) for the following DMA access. This should not be a problem if the DMA is used properly. The MD accesses are meant to stall the SDMA as little as possible to optimize throughput and hide calculation time. Therefore, final access to MS should be performed before closing a channel. This access waits until any pending operation is finished in the burst DMA and gather any remaining error.

In copy mode, an error could be immediately returned to the SDMA on execution of the `ldf` copy or `stf` copy instruction. It happens when MSA or MDA are not word addresses (for example, 0[4]). This is because copy mode must only be used for transferring a large packet of aligned data.

When an error is received during a *read* transfer to the external bus, which may occur during the burst accesses, the MD FIFO contains the valid beats of the burst, and the error flag of MS is set to 2'b11 (error read burst). It is possible to read MS ("n" field) to know how much valid data remains in MD and when MD is empty (after `ldf` instructions). The next read MD instruction sets the MS error flag to 2'b10 (error mode), and an error is sent back to the SDMA core. In error mode, it is possible to read MSA, which gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, gives rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

In "error read burst" mode, writing MDA, MSA, or MD, or starting a copy transfer by a `stf MD|COPY` instruction will cancel the error mode. Table 52-82 shows when an immediate error is sent back according to the executed instruction.

Table 52-82. Possibilities in ERROR READ BURST Mode

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA (|U |PF) stf rn, MDA stf rn,MD|COPY	NO	Error mode is reset. MSA, MDA, or MD are updated and a DMA cycle may start. For the `stf MD|COPY`, a copy loop is executed.
stf rn, MS	NO	MS is updated.

ldf rn, MS
ldf rn, MSA
ldf rn, MDA
	NO	MS, MSA, and MDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).

ldf rn, MD	YES/NO	Immediate error if there is no more data available for read in the FIFO.

When an error is received during a *write* transfer, the error is reported to the next DMA access. In this case, an error is sent to the SDMA core and the DMA goes to its error mode. Reading MS gives the number of bytes that remain in MD; reading MDA gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, give rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

Table 52-83. Possibilities in ERROR Mode

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA stf rn, MDA	Yes	Any attempt to modify MD, MSA, MDA will raise an immediate error and burst DMA remains in error mode. When address registers are write-accessed, an error is returned.
stf rn, MS	No	This is the only way to exit error mode. MS[9:8] must be reset by an `stf MS|SZ0` instruction.

ldf rn, MS
ldf rn, MSA
ldf rn, MDA
	No	MS, MSA, and MDA could be read in error mode without any side effects (for example, no DMA cycle is triggered).

ldf rn, MD	Yes	Whatever the DMA direction (read or write), an `ldf rn` triggers an immediate error.

52.18.1.11 Conditional Yielding

The standard SDMA transfer is based upon a hardware loop that has the following structure:

Example 52-2. Hardware Loop

```
loop
load Rn,source  // can be ldf or ld
<computation>   // can be done through functional units
store Rn,dest   // can be st or stf
done 0          // yield
```

This structure needs to be kept independent of the functional units' particularities regarding the context switch. However, there can be variations in the context switch's efficiency, which can depend on the number of data received up to that point, and on the data itself.

The DMA, with its 8-word burst capability, has a preferable context switch period when its address register is 8-word aligned: It is the only moment that occurs once every eight loops when the succession of bursts is not broken by the context switch. When this is not the case, a context switch requires the storing (or loading) of less than eight words, which requires separate accesses and is far less efficient. The rest of the 8-word packet is stored (or loaded) after the context restore, and this is done as separate accesses.

The proposed solution is a conditional yielding, which occurs only when the DMA is in an optimum state. It does not require any modification to the scripts. The condition is decided at the DMA level.

The DMA can be programmed in two modes—conditional or always-true—for every channel, which provides complete flexibility. By default, the DMA is not in conditional mode.

The DMA condition is computed from the FIFO fill level and the various modes, as follows:

- When copy mode is selected, regardless of the transfer direction ('read' or 'write'), the condition is always true.
- In read mode, the condition is always true.

- In write mode, the condition is true when there are four bytes or less in the FIFO; it is false when there are more than four bytes. The 4-byte limit comes from the possibility of saving those bytes as MD with absolutely no impact on the bus accesses.

The aim at conditional yielding is to avoid splitting bus accesses (especially bursts).

52.18.2 Burst DMA2 Unit

The DMA instructions control the DMA state machine and may cause a DMA cycle on the associated memory bus. There are four registers associated with the Burst DMA2 unit, a Memory Source Address register (DSA), a Memory Destination Address register (DDA), a Memory Data buffer (DD), and a state register (DS).
The burst DMA2 has two different uses:

- A data transfer between External Memory Interface and SDMA general register
- A data transfer in copy mode where blocks of data are transferred from the source address to the destination address

52.18.2.1 Memory Source Address Register (DSA)

The source address register contains the pointer into EMI memory associated with the next read data transfer. It has byte granularity.

Reading the register with the `ldf` instruction has no side effects, and gives the address value in the EMI memory of the next data that is read by the SDMA during an `ldf DD` instruction.

Writing the source address register has two side effects: If the prefetch bit is set, a DMA read cycle (8-word read access) is issued with the new address. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying DSA to guarantee all the data is effectively written to memory.

The DSA register has two modes of programming:

- Frozen—In frozen mode, the DSA register is not modified after DMA accesses.
- Incremented (default mode)—In incremental mode, DSA is incremented by the number of bytes transferred during read cycles.

52.18.2.2 Memory Destination Address Register (DDA)

The destination address register contains the pointer into EMI memory associated with the next write data transfer. It has byte granularity.

Reading the DDA register with the `ldf` instruction has no side effects. It gives the address value in the EMI memory where the next SDMA data (`stf r,DD` instruction) is stored when DD FIFO is flushed.

Writing the destination address register has one side effect. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying DDA to guarantee all the data is effectively written to memory.

The DDA register has two modes of programming:

- Frozen—In frozen mode, the DDA register is not modified after DMA accesses.
- Incremented (default mode)—The DDA register is incremented by the number of bytes transferred during write cycles.

52.18.2.3 Memory Data Buffer Register (DD)

The data buffer register consists of a bank of 36 bytes that behave like FIFO. This FIFO stores the eight words received when a read burst is triggered by the DMA (DMA is in read mode). The DD register is in write mode after a writing in DDA or after an `stf DD` instruction. In that case, a burst write access is automatically triggered when there are more than eight words in DD. For bandwidth optimization, any transfers between DMA and the EMI controller are based on burst accesses.

An `ldf r,DD|SIZE` instruction that reads the data buffer may cause a DMA cycle, as follows:

- If there are less bytes in the FIFO than the size parameter of the instruction. For instance, if only two bytes are available in DD and a 4-byte read is requested, a burst read access is executed to complete the two bytes.
- If the prefetch bit is set, and after reading there is enough space in the FIFO to store a full burst, a burst read access is triggered.

An `stf r,DD|SIZE` instruction that writes to the data buffer may cause a DMA cycle if the number of written bytes in DD is higher than 32 (eight words) or if the flush bit is set.

When DMA is used for data transfer between SDMA and EMI (reading or writing), no immediate error is possible because the module manages a data misalignment issue; therefore, it is allowed to read/write a word to/from a half-word address. However, the addresses (source or destination) must belong to the EMI memory mapping. The only potential error, in this mode, would be the error sent back by the EMI controller when an access to a super-user page is detected. The whole transfer on the DMA associated bus will be considered successful when there are no errors seen on the bus during the transfer. In copy mode, an immediate error could be returned to SDMA as described in Section 52.18.1.10, Error Management."

52.18.2.4 State Register (DS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer. DS is also accessed to set-up the conditional yielding feature.

The initialization value of this register is 0 and it consists of the following:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	stype	0	0	0	dtype
W																
R	0	0	0	0	y	d	e		0	0	n					
W																

Figure 52-71. DS Structure

Table 52-84. MS Field Descriptions

Field	Description
31-21	Reserved
20 stype	Source Mode. Indicates if DSA has to be incremented (or not) during accesses. 0 Frozen—DSA is not modified. 1 Incremented—DSA is incremented by the number of transferred bytes during read access.
19-17	Reserved
16 dtype	Destination Mode. Indicates if DDA has to be incremented (or not) during accesses. 0 Frozen—DDA is not modified. 1 Incremented—DDA is incremented by the number of transferred bytes during write access.
15–12	Reserved
11 y	Conditional Yielding selector. When selected, the `yield`/`yieldge` instructions will not switch channels if the Burst DMA2 is in Write Mode, and it has less than four bytes in its FIFO. This is aimed at reducing the number of inefficient FIFO flushes due to context switches. 0 Always yields 1 Yields conditionally (when there are less than four bytes in the FIFO in write mode)
10 d	Access Direction or DMA Mode. DMA is in write mode when data was written into DD by `stf DD` instructions, or if a previous DMA cycle on the external bus was a write access. Writing DDA or DSA changes the DMA mode to the respective value. DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by SDMA with an `ldf DD` instruction. Reading DDA or DSA does not change the DMA mode. 0 Read Mode 1 Write Mode
9–8 e	Error. Indicates if the previous access was acknowledged with a bus error. 00 No error was received. 01 *reserved* 10 Error mode 11 error read burst
7–6	Reserved
5–0 n	Number of bytes in the DD FIFO.

52.18.2.5 Burst DMA2 Write (stf)

When received from a `stf` instruction, the function code bits are interpreted as follows, depending on the addressed register:

Register	7	6	5	4	3	2	1	0
DSA	s		p	freeze	r			
DDA								
DD			f	cpy			sz	
DS								

Figure 52-72. STF Code Bits

Table 52-85. STF Code Bit Field Descriptions

Field	Description
7–6 s	Functional Unit selector 00 for Burst DMA2
5 p (DSA)	Prefetch Flag 0 No prefetch 1 Prefetch required from new DSA
5 f (DD)	Forced Flush Flag 0 Automatic flush 1 FIFO contents are flushed (including the new written data).
4 freeze (DSA/DDA)	Address Freeze Mode 0 Address is normally incremented. 1 Address is frozen.
4 cpy (DD)	Copy Mode selection 0 Write Mode 1 Copy Mode
3–2 r	Register selection 00 DSA 01 DDA 10 DD 11 DS
1–0 sz (DD/DS)	Transfer Size 00 size 0 (no data stored in the FIFO) 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)

The possible write instructions are listed in Table 52-77 (unused bits should always be cleared).

Table 52-86. Burst DMA2 STF Instruction List

Binary	Assembly	Comments
01_0_0_00_00	stf r,DSA	Writes content of the SDMA general register (r) to the source address register. DSA is in incremented mode.
01_0_1_00_00	stf r,DSA|FR	Writes content of the SDMA general register (r) to the source address register. DSA is in frozen mode.
01_1_0_00_00	stf r,DSA|PF	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access. DSA is in incremented mode.

Table 52-86. Burst DMA2 STF Instruction List (continued)

Binary	Assembly	Comments
01_1_1_00_00	stf r,DSA|PF|FR	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access.
01_0_0_01_00	stf r,DDA	Writes content of the SDMA general register (r) to the destination address register. DDA is in incremented mode.
01_0_1_01_00	stf r,DDA|FR	Writes content of the SDMA general register (r) to the destination address register. DDA is in frozen mode.
01_1_0_10_00	stf r,DD|SZ0|FL	No data transfers between the SDMA and DD, but all valid written data of the DD is flushed to the memory. An acknowledge or error is sent back to the SDMA core on transfer completion.
01_0_0_10_01	stf r,DD|SZ8	8-bit (byte) transfer to write buffer DD
01_1_0_10_01	stf r,DD|SZ8|FL	8-bit (byte) transfer to write buffer DD and flush after transfer. All valid written data of the DD is flushed to memory.
01_0_0_10_10	stf r,DD|SZ16	16-bit (half-word) transfer to write buffer DD
01_1_0_10_10	stf r,DD|SZ16|FL	16-bit (half-word) transfer to write buffer DD and flush after transfer. All valid written data of the DD is flushed to memory.
01_0_0_10_11	stf r,DD|SZ32	32-bit (word) transfer to write buffer DD
01_1_0_10_11	stf r,DD|SZ32|FL	32-bit (word) transfer to write buffer DD and flush after transfer. All valid written data of DD is flushed to memory.
01_0_1_10_00	stf r,DD|CPY	No data transfer between SDMA and DD but starts a copy transfer whose length is given by the 4 LSB of *r* register. (Maximum burst length is eight words.)
01_0_0_11_11	stf r,DS	32-bit (word) transfer to status register DS
01_0_0_11_00	stf r,DS|SZ0	Clears the error flag (if set). Other DS bits are unchanged; this instruction is also known as `clref` DS.

NOTE

When a flush bit is set, the SDMA flushes the FIFO including the newly written data. An acknowledge is sent to the core before the flush completes (except if size 0 is used). The goal of this flush bit is to force a flush, but it is recommended to use it only when needed (for example, when finishing a row of pixels during 2D data transfers). Indeed, if this bit is omitted and if there are more than 32 bytes in the FIFO, a burst write access is automatically triggered.

Since all the `stf r,DD` instructions (including the copy mode) acknowledge the SDMA core before the store is effective (except if size 0 is used), it is recommended to perform an `ldf` from DS before terminating a channel in order to check the final error status. (The `ldf` from DS will stall the core until all the data was flushed out and the transfer status is known.)

After every `stf DD` instruction, the DDA is incremented by the number of bytes that are written in DD, except when it is programmed in frozen mode.

52.18.2.6 Burst DMA2 Read (ldf)

When received from an `ldf` instruction, the function code bits are interpreted as follows, depending on the addressed register:

Register	7	6	5	4	3	2	1	0
DSA	s				r			
DDA								
DD			p				sz	
DS								

Figure 52-73. LDF Code Bits

Table 52-87. LDF Code Bit Field Descriptions

Field	Description
7–6 s	Functional Unit selector 00 for Burst DMA2
5 p (DD)	Prefetch Flag 0 no prefetch 1 automatic prefetch
3–2 r	Register selection 00 DSA 01 DDA 10 DD 11 DS
1–0 sz (DD)	Transfer Size 00 reserved 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)

Table 52-79 lists the possible write instructions (unused bits should always be cleared).

Table 52-88. Burst DMA2 LD2F Instruction List

Binary	Assembly	Comments
01_0_0_00_00	ldf r,DSA	Copies the source address register value into an SDMA general register. It gives the memory address of the next data that will be read with an `ldf` DD instruction.
01_0_0_01_00	ldf r,DDA	Copies the destination address register value into an SDMA general register. It gives the memory address where the next incoming data will be flushed.
01_0_0_10_01	ldf r,DD|SZ8	8-bit (byte) read
01_1_0_10_01	ldf r,DD|SZ8|PF	8-bit (byte) read. If after this reading and the DD FIFO is empty, a burst read access at the DSA address is triggered.
01_0_0_10_10	ldf r,DD|SZ16	16-bit (half-word) read

Table 52-88. Burst DMA2 LD2F Instruction List (continued)

Binary	Assembly	Comments
01_1_0_10_10	ldf r,DD|SZ16|PF	16-bit (half-word) read. If after this reading, and the DD FIFO is empty, a burst read access at the DSA address is triggered.
01_0_0_10_11	ldf r,DD|SZ32	32-bit (word) read
01_1_0_10_11	ldf r,DD|SZ32|PF	32-bit (word) read. If after this reading and the DD FIFO is empty, a burst read access at the DSA address is triggered.
01_0_0_11_00	ldf r,DS	Copy the status register value into an SDMA general register.

NOTE

Read data is 0-extended before writing in the SDMA general registers. When reading the DD register, the DMA takes data from the FIFO if it is available. If part or whole data is not in the FIFO, an external burst read access is performed to provide the missing data. The SDMA is stalled as long as the required read data is not complete.

After every reading, DSA is incremented by the number of read bytes from DD FIFO, except when DSA is programmed in frozen mode.

52.18.2.7 Prefetch/Flush and Auto-Flush Management

The prefetch and auto-flush management enables the SDMA RISC machine to go on while a DMA access is performed. When the RISC core requires a prefetch (p = 1) to the Burst DMA2, it will receive an immediate transfer acknowledge before the DMA has finished the external access. This enables the RISC core to do other things like accessing another DMA machine.

The basic principle in prefetch mode is for the DMA to anticipate data reads from the SDMA RISC engine by fetching external bursts of data as soon as there is enough space in the DMA FIFO to store it. If ever the RISC engine required data that is not available in the FIFO, the read acknowledge is delayed until the data is available, but it does not have to wait until the burst completes.

The auto-flush basic principle is similar: An automatic flush is triggered every time there are eight words to be written in the FIFO. If the FIFO is full and the RISC engine requires another write, it is stalled until the burst has started and enough space was freed in the FIFO to store that new data. This means the SDMA RISC engine does not have to wait for the completion of a burst to receive its acknowledge and continue its processing.

In particular, an auto-flush is executed when DMA is in write mode and if the following is true:

- If the FIFO is empty and the first write is to a word-aligned address of any size (ex: the 2 LSB of DDA[1:0]= 0x0), the auto-flush is triggered immediately after the write of the 32’nd byte.
- If the FIFO is empty, and if DDA is an odd byte address (1, 3, 5, 7, . . .) and an `stf DD|SZ8` is executed, the byte is flushed to memory. Once DDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.

- If the FIFO is empty, and if DDA is a half-word address (2, 6, 0xA, . . .) and an `stf DD|SZ16` is executed, the two bytes of the incoming data are flushed to memory. Once DDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if DDA is not a word-aligned address (ex 1, 2, 3, 5, 6, 7, 9, . . .), and an `stf DD|SZ32` is executed, the first 1 to 3 bytes will be flushed up to the next word aligned address. Afterwards, an auto-flush will be triggered each time the FIFO receives 32-bytes.

  Therefore, if an `stf DD|SZ32` is executed with DDA equal to 0x1 and with an empty DD FIFO, the bytes located at addresses 1, 2, and 3 are flushed, and the byte located at address 4 remains in DD FIFO. This solves the misalignment issue. Additionally, the next write instructions (`stf`) complete the FIFO until it contains eight words; then a burst write is executed by the DMA to empty the FIFO. Protocol on the external bus does not support bursts of different data types (byte, half-word, or word).

For example, consider the case where data is written using a byte access, `stf DD|SZ8`. The value of DDA during the very first byte write determines when the auto-flush will occur as follows:

- If DDA=0x0, the flush occurs following the write of byte 32
- If DDA=0x1, the flush occurs following the write of byte 1, byte 3 and byte 35.
- If DDA=0x2, the flush occurs following the write of byte 2 and byte 34.
- If DDA=0x3, the flush occurs following the write of byte 1 and byte 33.
- If DDA=0x4, the flush occurs following the write of byte 32

The flush command forces the DMA to flush all DD valid bytes to the EMI controller. An acknowledge is sent immediately to the SDMA, and any potential error is reported on a future access. It is thus essential to conclude a transfer with a last read from DS, which will stall the core until all data was flushed out and returned to the transfer status (acknowledge or error).

NOTE

During this kind of auto-flush—which occurs only at the beginning of a misaligned write transfer—no acknowledge is sent back to the SDMA, which is stalled until a flush is completed.

52.18.2.8 Data Alignment and Endianness

52.18.2.8.1 Burst DMA2 in Read Mode

For every read access to DD, the data returned to the SDMA core and the new FIFO state depends on the DSA status and the access size. The FIFO is considered as a stack of 36 bytes: Data is fetched externally on a 32-bit bus, but the valid bytes only are stored in the FIFO and left-aligned (for a transfer of consecutive words, it is only the first word that may be truncated). Table 52-80 shows the FIFO byte alignment strategy and the corresponding DSA, the returned data, and the new FIFO state for any access size of an internal read from DD.

Table 52-89. FIFO Read Configuration

Before read		Internal read access size	Read data	After read	
DSA[1:0]	FIFO state			DSA[1:0]	FIFO state
00	x0 x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 and so on...	sz8	00 00 00 x0	01	x1 x2 x3 y0 y1 y2 y3 z0
		sz16	00 00 x0 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz32	x0 x1 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
01	x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 and so on...	sz8	00 00 00 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz16	00 00 x1 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz32	x1 x2 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
10	x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 and so on...	sz8	00 00 00 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz16	00 00 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz32	x2 x3 y0 y1	10	y2 y3 z0 z1 z2 z3 t0 t1
11	x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 t2 and so on...	sz8	00 00 00 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz16	00 00 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
		sz32	x3 y0 y1 y2	11	y3 z0 z1 z2 z3 t0 t1 t2

52.18.2.8.2 Burst DMA2 in Write Mode

For every write access to the DD, the new FIFO state depends on the DDA status and the access size. The FIFO is considered as a stack of 36 bytes: Data is stored in the FIFO according to the internal access size

and the former DDA value. Table 52-81 shows the FIFO byte alignment strategy corresponding to DDA, as well as the new FIFO state for any access size of an internal write to DD.

Table 52-90. FIFO Write Configuration

Before write		Internal write access size	Written data	After write	
DDA[1:0]	FIFO state			DDA[1:0]	FIFO state
00	tt uu vv ww ?? ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	01	tt uu vv ww x0 ?? ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	10	tt uu vv ww x0 x1 ?? ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	00	tt uu vv ww x0 x1 x2 x3 ?? ?? ?? ??
01	tt uu vv ww xx ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	10	tt uu vv ww xx x0 ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	11	tt uu vv ww xx x0 x1 ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	01	tt uu vv ww xx x0 x1 x2 x3 ?? ?? ??
10	tt uu vv ww xx yy ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	11	tt uu vv ww xx yy x0 ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	00	tt uu vv ww xx yy x0 x1 ?? ?? ?? ??
		sz32	x0 x1 x2 x3	10	tt uu vv ww xx yy x0 x1 x2 x3 ?? ??
11	tt uu vv ww xx yy zz ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	00	tt uu vv ww xx yy zz x0 ?? ?? ?? ??
		sz16	?? ?? x0 x1	01	tt uu vv ww xx yy zz x0 x1 ?? ?? ??
		sz32	x0 x1 x2 x3	11	tt uu vv ww xx yy zz x0 x1 x2 x3 ??

NOTE

If the FIFO mode changes from a write to a read mode, all remaining written bytes in DD are lost but no error is returned. Typically, this happens if an `ldf` DD is executed after `stf` DD instructions. Before a mode change, it is recommended to force the flush of a potential remaining byte by a `stf DD|SZ0|FL` instruction. In the same way, if a FIFO mode changes from a read to a write mode, all prefetched data present in the FIFO is lost and no error is returned.

52.18.2.8.3 Endianness

Big and Little Endian are supported by the Burst DMA2, but data is always stored in DD in Big Endian. Byte manipulation is performed when data is exchanged with an Burst controller (for example, during read or write burst accesses).

52.18.2.9 Copy Mode

A mechanism is available to perform fast AP-to-AP transfers. Data does not flow through the SDMA core: It is kept in the DMA FIFO. This mechanism is selected when writing DD with a special option in the instruction code (copy flag). It is possible to transfer up to eight words in one SDMA instruction (this does not mean in one cycle). In this mode, every time an `stf DD|CPY` is executed, a read burst is executed and directly followed by a write burst transfer. Burst transfers are limited to eight words. The size of the transfer (in words)—given by the SDMA general register (4 LSB)—is also limited to eight. The following SDMA code shows how 100 bytes could be copied from the DSA address to the DDA address. This is sample code only.

Example 52-3. Burst DMA2 copy mode example

```
        ldi r0,@src
        stf r0,DSA                  // Source address setup
        ldi r1,@dst
        stf r1,DSA                  // Destination address setup
        ldi r0,0x64                 // data transfer counter
        ldi r1,0x8
MAIN_XFER:
        cmphs r0,r1                 // Is r0 >= 0x8
        bf LAST_XFER                // If not, jump to last transfer label
        stf r1,DD|CPY               // Copy 8 words from DSA to DDA address.
        subi r0,0x8                 // Decrement counter
        jmp MAIN_XFER               // return to main transfer loop
LAST_XFER:
        stf r0,DD|CPY
```

The main transfer loop is executed 12 times; then `r0` equals 4 and the last transfer loop is run.

In this mode, an acknowledge is transmitted to the core as soon as the read burst can start; thus, a first copy instruction returns an immediate acknowledge and subsequent copy instructions will be acknowledged as soon as the previous copy has finished.

52.18.2.10 Error Management

Another point to consider is the management of errors. Because the DMA immediately sends an acknowledge to the RISC core (except for the `stf DS|SZ0|FLS` instruction), it assumes no error will occur. If an error occurs, it is flagged (transfer error acknowledge) for the following DMA access. This should not be a problem if the DMA is used properly. The DD accesses are meant to stall the SDMA as little as possible to optimize throughput and hide calculation time. Therefore, final access to DS should be performed before closing a channel. This access waits until any pending operation is finished in the Burst DMA2 and gather any remaining error.

In copy mode, an error could be immediately returned to the SDMA on execution of the `ldf` copy or `stf` copy instruction. It happens when DSA or DDA are not word addresses (for example, 0[4]). This is because copy mode must only be used for transferring a large packet of aligned data.

When an error is received during a *read* transfer to the external bus, which may occur during the burst accesses, the DD FIFO contains the valid beats of the burst, and the error flag of DS is set to 2'b11 (error read burst). It is possible to read DS ("n" field) to know how much valid data remains in DD and when DD is empty (after `ldf` instructions). The next read DD instruction sets the DS error flag to 2'b10 (error mode), and an error is sent back to the SDMA core. In error mode, it is possible to read DSA, which gives the address of the error data. Any attempt to read or write DD, or to modify DDA or DSA in error mode, gives rise to an error; therefore, an error flag must be reset by clearing DS at the end of the SDMA code section responsible for error management.

In "error read burst" mode, writing DDA, DSA, or DD, or starting a copy transfer by a `stf DD|COPY` instruction will cancel the error mode. Table 52-82 shows when an immediate error is sent back according to the executed instruction.

Table 52-91. Possibilities in ERROR READ BURST Mode

DMA Instruction	Immediate Error	Comments
stf rn, DD stf rn, DSA (IU IPF) stf rn, DDA stf rn,DD|COPY	NO	Error mode is reset. DSA, DDA, or DD are updated and a DMA cycle may start. For the `stf DD|COPY`, a copy loop is executed.
stf rn, DS	NO	DS is updated.

ldf rn, DS
ldf rn, DSA
ldf rn, DDA
	NO	DS, DSA, and DDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).

ldf rn, DD	YES/NO	Immediate error if there is no more data available for read in the FIFO.

When an error is received during a *write* transfer, the error is reported to the next DMA access. In this case, an error is sent to the SDMA core and the DMA goes to its error mode. Reading DS gives the number of bytes that remain in DD; reading DDA gives the address of the error data. Any attempt to read or write DD, or to modify DDA or DSA in error mode, give rise to an error; therefore, an error flag must be reset by clearing DS at the end of the SDMA code section responsible for error management.

Table 52-92. Possibilities in ERROR Mode

DMA Instruction	Immediate Error	Comments
stf rn, DD stf rn, DSA stf rn, DDA	Yes	Any attempt to modify DD, DSA, DDA will raise an immediate error and Burst DMA2 remains in error mode. When address registers are write-accessed, an error is returned.
stf rn, DS	No	This is the only way to exit error mode. DS[9:8] must be reset by an `stf DS|SZ0` instruction.

ldf rn, DS
ldf rn, DSA
ldf rn, DDA
	No	DS, DSA, and DDA could be read in error mode without any side effects (for example, no DMA cycle is triggered).

ldf rn, DD	Yes	Whatever the DMA direction (read or write), an `ldf rn` triggers an immediate error.

52.18.2.11 Conditional Yielding

The standard SDMA transfer is based upon a hardware loop that has the following structure:

Example 52-4. Hardware Loop

```
loop
load Rn,source  // can be ldf or ld
<computation>   // can be done through functional units
store Rn,dest   // can be st or stf
done 0          // yield
```

This structure needs to be kept independent of the functional units' particularities regarding the context switch. However, there can be variations in the context switch's efficiency, which can depend on the number of data received up to that point, and on the data itself.

The DMA, with its 8-word burst capability, has a preferable context switch period when its address register is 8-word aligned: It is the only moment that occurs once every eight loops when the succession of bursts is not broken by the context switch. When this is not the case, a context switch requires the storing (or loading) of less than eight words, which requires separate accesses and is far less efficient. The rest of the 8-word packet is stored (or loaded) after the context restore, and this is done as separate accesses.

The proposed solution is a conditional yielding, which occurs only when the DMA is in an optimum state. It does not require any modification to the scripts. The condition is decided at the DMA level.

The DMA can be programmed in two modes—conditional or always-true—for every channel, which provides complete flexibility. By default, the DMA is not in conditional mode.

The DMA condition is computed from the FIFO fill level and the various modes, as follows:

- When copy mode is selected, regardless of the transfer direction ('read' or 'write'), the condition is always true.
- In read mode, the condition is always true.

- In write mode, the condition is true when there are four bytes or less in the FIFO; it is false when there are more than four bytes. The 4-byte limit comes from the possibility of saving those bytes as DD with absolutely no impact on the bus accesses.

The aim at conditional yielding is to avoid splitting bus accesses (especially bursts).

52.18.3 OnCE and Real-Time Debug

The On-Chip Emulation module (OnCE) is the debug interface to the SDMA. It supports the access to all core internal devices (registers, memory, and so on), and provides a set of mechanisms that control the core. The OnCE is accessed by JTAG ports at the chip's board level, or by the host via its peripheral bus.

To reduce the size of the hardware material involved, all tasks supported by the OnCE are performed on the SDMA core. The architecture of the SDMA OnCE is relatively simple and very flexible.

The commands supported by the SDMA OnCE are listed in the following sections.

52.18.3.1 Memory and Register Access

A set of mechanisms is provided to access SDMA memory and register locations. Both reading and writing are allowed. The access is supported if the processor is in debug mode.

Those registers can also be accessed through the AP Control interface when the OnCE is controlled by the AP, as described in Section "Using the BP."

52.18.3.2 Hardware Breakpoints

An event detection unit is implemented to support memory breakpoints. The unit watches the data exchanged between the SDMA memory bus and the core. A debug request is sent to the core when matching conditions occur. The unit supports mixed conditions based on address range, access type, and data value. Event detection unit configuration registers are memory mapped in the SDMA space (see Section 52.14.3, SDMA Core Register Descriptions"): You can modify them through a regular memory access or the AP control interface.

52.18.3.3 Watchpoints

One output pin is provided to monitor matching trigger conditions that are defined in the event detection unit.

52.18.3.4 Software Breakpoints

The SDMA instruction set contains a software breakpoint. Upon executing a software breakpoint instruction, the core suspends normal execution and enters debug mode. No hardware step execution mode is implemented in the OnCE module, but this feature may be implemented at the software level with this instruction.

52.18.3.5 Core Control

Commands are provided to monitor and control processor activity. You can halt the core, rerun the core from another address location, and get processor status. Any hardware breakpoint on the instruction bus is not supported, but this feature may be implemented by inserting a software breakpoints program.

52.19 The OnCE Controller

The OnCE controller receives commands from the AP or from the JTAG controller. Each command is interpreted before being sent to the core.

52.19.1 OnCE Commands

A small set of commands supports the communication between the OnCE module and the external world. This command set enables you to perform any of the following tasks: control processor activity, save core context, and execute an SDMA instruction from the OnCE module. Combined together, these tasks perform more complex commands.

A full OnCE command contains a 4-bit instruction (the OnCE command opcode) and a variable length data field (the OnCE data). During command execution, the OnCE data is transferred in a OnCE internal register before being exchanged with the SDMA. Some data values are also exported. This mechanism creates a link between the processor and the external world. Nine commands are defined: Table 52-93 presents their formats.

Table 52-93. OnCE Command Opcode Values

Instruction Opcode	Name	Action	Register	Data Field Size	Mode
0000	rstatus	Reads the OnCE status register	STATUS	16-bit	normal/debug
0001	dmov	Updates general register GReg1	GREG1	32-bit	debug
0010	exec_once	Runs the instruction from the SDMA instruction register	INSTRUCTION	16-bit	debug
0011	run_core	Returns to normal execution	BYPASS	1-bit	debug
0100	exec_core	Returns to normal execution via a jump instruction that specifies the new address	INSTRUCTION	16-bit	debug
0101	debug_rqst	Stops the core after execution of current instruction	BYPASS	1-bit	normal
0110	rbuffer	Reads the real time buffer	RTB	32-bit	normal/debug
0111-1110	reserved	Reserved	BYPASS	1-bit	normal/debug
1111	bypass	Bypasses TAP controller	BYPASS	1-bit	normal/debug

Each instruction corresponds to a specific action performed on the OnCE module. The nature of the associated data field is clearly identified. The `dmov` command is followed by a 32-bit data value (which is a data value for the SDMA); the `exec_once` and the `exec_core` commands are followed by a 16-bit data value (which is an instruction for the SDMA); the `rstatus` command is followed by a 16-bit control value (which is the content of the OnCE status register); the `rbuffer` command is followed by a 32-bit data value.

The debug_rqst and the run_core commands are followed by a single bit data field (this is a bypass value). Finally, the bypass instruction enables the SDMA JTAG TAP controller to be daisy-chained with another JTAG TAP controller. This is a JTAG-only feature. The set of commands is simple, but enables you to perform any possible task on the SDMA during a debug process.

52.19.2 Sending Commands to the OnCE Controller

The JTAG access is the standard access to the OnCE, but sometimes the JTAG is not available to fix some bugs (if the chip is in production for instance), an additional access is then required. Therefore, one AP access to the OnCE is provided.

52.19.2.1 Using the JTAG Interface

A serial access is performed through the five JTAG pins TCK, TRST, TMS, TDI, and TDO. A Test Access Port controller is provided to decode the TMS control signal. It produces shift-enable signals (shift_ir and shift_dr), and updates enable signals (update_ir and update_dr). It is fully compliant with the IEEE 1149.1 testability (JTAG) standard.

During the shift_ir state, the command opcode is shifted into the OnCE controller (for example, the signal from the TDI pin is shifted into the command register and the TDO pin receives the signal shifted out). After transferring the four bits of the command, an update_ir signal is asserted and the command is decoded. The target data register is now clearly identified and the corresponding control signal is produced, as follows: bypass enable signal (bp_en), instruction enable signal (inst_en), data enable (data_en), and status enable signal (stat_en).

During the shift_dr state, the TDI signal is shifted into one of the following target registers: bypass register (1 bit), SDMA instruction register (16 bits), SDMA data register (32 bits), or OnCE status register (16 bits). The TDO pin is connected to the output of the selected register to receive the signals shifted out.

The JTAG access is disabled when the AP access is enabled.

52.19.2.2 Using the AP

The AP access to the OnCE is not the standard access, but it is required if the JTAG is not available. For example, if the SDMA ROM is out of use on a chip in production, and the AP needs to download new code and restart the SDMA, the OnCE can easily perform this operation. This type of debug operation justifies the use of an AP access to the OnCE.

To drive the OnCE, the AP uses some registers contained in the AP Control module of the SDMA. These registers are accessed through the AP peripheral bus. Most of these registers are connected to another register in the OnCE controller. Thus, accessing one of these registers is equivalent to accessing the associated register in the OnCE controller.

The set of registers in the AP Control module is listed below:

- ONCE_ENB register (1 bit, read/write)—This 1-bit register enables the AP access to the OnCE. When this bit is set, the signals from the JTAG are ignored. When it is cleared, all writing operations to the following registers through the Host Control interface are ignored. This register is reset on a JTAG reset.

- ONCE_CMD register (4 bits, read/write)—This 4-bit register receives the command opcode. It is connected to the command register in the controller. A write access to this register causes the associated command to be executed on the OnCE. For example, after writing “0001” in this register, a `dmov` command is executed.

NOTE

On the AP side, the `rstatus` and `bypass` commands are not supported. This register is reset on a JTAG reset.

- ONCE_DATA register (32 bits, read/write)—This 32-bit register is connected to the SDMA data register. This register is used when executing a `dmov` or `rbuffer` command.

NOTE

Before requesting a `dmov` command, the 32-bit data to transfer must be written in the ONCE_DATA register. At the end of the execution, the register is updated with GReg1 former value. This register is reset on a JTAG reset.

- ONCE_INSTR register (16 bits, read/write)—This 16-bit register is connected to the SDMA instruction register. This register is used when executing an `exec_core` or an `exec_once` command.

NOTE

Before requesting an `exec_core` or an `exec_once` command, the appropriate instruction must be written in the ONCE_INSTR register. This register is reset on a JTAG reset.

- ONCE_STAT register (16 bits, read only)—A read access to the ONCE_STAT register returns the content of the OnCE status register (OSTAT). This register is read only.
- The bypass register is not useful when the AP controls the OnCE, therefore no register is defined in the AP Control module to access the bypass register.

52.19.2.3 Conflicts Between the JTAG and the AP Accesses

When AP access to the SDMA OnCE is enabled (that is, when the bit in the ONCE_ENB register is set), the JTAG access is disabled. This guarantees that the module is not accessed at the same time on both sides.

It is possible to check whether the JTAG access to the SDMA OnCE is enabled from the JTAG port. When the JTAG access is disabled, the SDMA TDO always returns 1. The check requires the following steps:

- Execute a `dmov` command from debug mode (with neither 0xffffffff nor 0x0 as `dmov` value: 0x5a5a5a5a is good).
- Execute another `dmov` command (the value here is not important).

  The returned value from the latter `dmov` command should be the original one if the JTAG access is enabled; if it is 0xffffffff instead of the original input value, this means the JTAG access is disabled.

52.19.3 Executing a Command from the OnCE

All the commands defined in Section 52.19.1, OnCE Commands" can be accessed through the JTAG. The AP can access all these commands except the `rstatus` command. On the AP side, the OnCE status is directly accessed by reading the ONCE_STAT register.

52.19.3.1 Nature of the Commands

Two types of commands may be distinguished. First, there are two commands that do not interact with the core: `rstatus` and `rbuffer`. Those commands may be requested at any time: They do not depend on the core status.

NOTE

Each of these commands exports a data value or a status value from the SDMA.

There are also commands that interact with the core: `dmov`, `run_core`, `exec_core`, `exec_once`, and `debug_rqst`. These commands are core status dependent, as follows:

- During user mode only the `debug_rqst` is taken into account.
- During debug mode, all these commands are taken into account except the `debug_rqst`. For example, an `exec_once` command requested while not in debug mode has no effect.

52.19.3.2 Execution Request

The SDMA starts executing a task in debug mode when requested by the OnCE controller. The execution starting time depends on the type of access used to communicate with the OnCE.

If the JTAG is used, the request is send after decoding the `update_dr` state in the TAP controller. Therefore, always cross this state when sending a command through the JTAG. If the OnCE is driven from the AP side, the request is sent after detecting a write access to the ONCE_CMD register. All the registers involved in this operation must be loaded first.

The following is an example of an `exec_core` command execution from the AP side: After writing '010' in the ONCE_CMD register, the OnCE controller asks the SDMA to execute the instruction contained in the ONCE_INSTR register. The instruction involved should be available in the ONCE_INSTR register before the beginning of the execution.

52.19.3.3 Command Execution

The following list shows the commands and details how each command is executed:

- `rstatus` command execution—The `rstatus` command exports the content of the OnCE status register (OSR). If the JTAG is used, the status information is captured in the OnCE status register during the capture_dr state, and shifted out after 16 TCK clock cycles in the shift_dr state. The `rstatus` command is not supported on the AP side, but a status register is provided instead. The `rstatus` may be performed in both debug and user modes.

- `dmov` command execution—The `dmov` command accesses SDMA internal registers. Executing a `dmov` instruction exchanges the 32-bit data values between the SDMA data register and the general register GReg[1].

  If the JTAG is used, the content of GReg1 is captured in the SDMA data register during the capture_dr state, then it is shifted out after 32 TCK clock cycles in the shift_dr state. During the update_dr state, GReg1 is updated with the new, shifted-in 32-bit data value. If the OnCE is driven from the AP side, the data values contained in GReg1 and the SDMA data register are exchanged after detecting a write access to the ONCE_CMD register. The ONCE_DATA register must therefore be loaded first.

- `exec_once` command execution—The `exec_once` command executes the instruction loaded in the SDMA instruction register. The command may only be requested from debug mode. The SDMA returns to debug mode at the end of the execution.

  Change of flow instructions as well as instructions that may cause a context switch are not supported: The comprehensive list comprises `done/yield/yiedge` (except `done` 5), BF, BT, BSF, BDF, JMP, JSR, JMPR, JSRR, RET, and LOOP, as well as all the illegal instructions.

  No other command should be requested before the SDMA returns to debug mode. The SDMA status (for example, whether it is in debug mode or not) can be detected by polling with the `rstatus` OnCE command, monitoring the debug_mode pin, or checking the Section 52.12.3.19, OnCE Status Register (ONCE_STAT)" register via the AP control interface.

  NOTE

  Most of the instructions are single-cycle, which omits the step of polling the status. Loads and stores to DMA units are typical instructions that might require this polling.

  If the JTAG is used, the 16-bit instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift_dr state. A request is sent to the core when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the AP side, the request is sent to the SDMA when detecting a write access to the ONCE_CMD register. The ONCE_INSTR register must be therefore be loaded first.

- `run_core` command execution—The `run_core` command leaves debug mode and resume normal program execution. The next instruction executed is the last instruction decoded before entering debug mode. Be sure to restore core context before re-running the core. This procedure is detailed in Section 52.20.5.3, Restoring the Context."

  If the JTAG is used, a 1-bit bypass value is shifted in the bypass register in the shift_dr state. The SDMA is rerun when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the AP side, the core is rerun when detecting a write access to the ONCE_CMD register.

- `exec_core` command execution—The `exec_core` command resumes program execution from any address. The 16-bit instruction provided with the `exec_core` overwrites the last instruction decoded before entering debug mode. This command is designed to support change of flow instructions, so that a program execution can be restarted from any address. After executing an `exec_core` command, the SDMA leaves debug mode. The `exec_core` command is usually used with a `jmp` instruction.

If the JTAG is used, the 16-bit branch instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift_dr state. The SDMA is rerun when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the AP side, the SDMA reruns when detecting a write access to the ONCE_CMD register. The ONCE_INSTR register must therefore be loaded first. For example, to restart the SDMA from the program address 0x100, the instruction loaded should be a `jump to address 0x100` instruction.

- `debug_rqst` command execution—The `debug_rqst` command puts the SDMA in debug mode. If the JTAG is used, a 1-bit bypass value is shifted in the bypass register during the shift_dr state. A debug request is sent to the SDMA when the update_dr state is decoded in the TAP controller. If the OnCE is driven from the AP side, the debug request is sent when detecting a write access to the ONCE_CMD register. When the SDMA is already in debug mode, this command is simply ignored.
- `rbuffer` command execution—The `rbuffer` command exports the content of the real time buffer (RTB). If the JTAG is used, the content of the real time buffer (RTB) is captured in the SDMA data register during the capture_dr state. The register is completely shifted out after maintaining the shift_dr state during 32 TCK clock cycles. If the OnCE is driven from the AP side, the content of the RTB is captured in the ONCE_DATA register after detecting a write access to the ONCE_CMD register.
- `bypass` command execution—This command is only available from the JTAG interface. It enables daisy-chaining of the SDMA JTAG TAP controller with other JTAG TAP controllers. This command does not change the SDMA state and can be executed in any mode (run, debug, or sleep). It selects the bypass register of the TAP controller.

52.19.4 Registers Descriptions

See Section 52.14.3, SDMA Core Register Descriptions," and Section 52.12, AP Memory Map and Control Register Definitions," for detailed information on each register.

52.19.4.1 Event Cell Counter Register (ECOUNT)

The event cell counter register is a 16-bit register that contains the number of times minus one that an event detection occurs before generating a debug request. This register should be written before attempting to use the event detection counter during an event detection process. The event cell counter register is cleared on a JTAG reset.

52.19.4.2 Event Cell Address Registers (EAA or EAB)

The event cell contains two address registers—the event cell address register (a), called EAA, and the event cell address register (b), called EAB. Every address register is a 16-bit register that stores a user-defined address value. This value computes one of the following address conditions: addra_cond or addrb_cond. Every address register is cleared on a JTAG reset.

52.19.4.3 Event Cell Address Mask Register (EAM)

The event cell address mask register is a 16-bit register that contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before comparing addresses.

NOTE

There is a common address mask value for the two address comparators. If bit *i* of this register is set, then bit *i* of the address value latched from the memory bus does not influence the result of the address comparison. The event cell address mask register is cleared on a JTAG reset.

52.19.4.4 Event Cell Data Register (ED)

The event cell data register is a 32-bit register that contains a user-defined data value. This data value is an input for the data comparator, which generates the data_cond condition. The event cell data register is cleared on a JTAG reset.

52.19.4.5 Event Cell Data Mask Register (EDM)

The event cell data mask register is a 32-bit register that contains a user-defined data mask value. This mask is applied to the data value latched from the memory bus before comparing data. Setting bit *i* of the event cell data mask register means that bit *i* of the data value latched from the address bus does not influence the result of the data comparison. The event cell data mask register is cleared on a JTAG reset.

52.19.4.6 Real Time Buffer Register (RTB)

The real Time Buffer register is a 32-bit register that stores and retrieves run-time information without putting the SDMA in debug mode. Refer to Section 52.20.8.2, Real Time Buffer" for more details.

52.19.4.7 Event Control Register (ECTL)

The event cell control register is a 16-bit register that defines cell event occurrence conditions. The event cell control register is cleared on a JTAG reset. See also Section 52.20.6, OnCE Event Detection Unit" for more details.

52.19.4.8 Trace Buffer (TB)

The Trace Buffer register retrieves the information in the Trace Buffer. See Section 52.20.8.1, Trace Buffer" for more details.

52.19.4.9 OnCE Status Register (OSTAT)

The OnCE status register is a 16-bit register that contains processor and event detection unit status. The OSTAT is a read-only register. Refer to Section 52.12.3.19, OnCE Status Register (ONCE_STAT)" for detailed description of the individual fields in the OSTAT register.

Figure 52-74 shows the OSTAT structure.

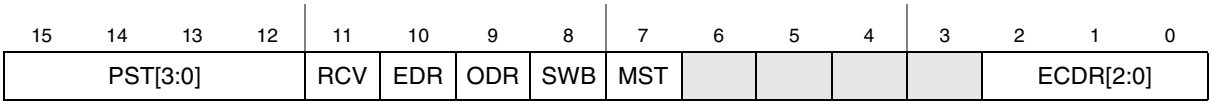


Figure 52-74. OnCE Status Register (OnCE)

Where PST[3:0] is the SDMA core state, RCV is set when the real-time buffer (RTB) is modified. EDR, ODR, and SWB are set, respectively, when the SDMA has entered debug mode because of an external debug request, a OnCE debug_rqst command, or a software breakpoint. MST is set when the OnCE is controlled from the AP control interface, and when ECDR is a three-flag set that shows the event cell condition(s) that put the core in debug mode. The OSTAT never provides more than one reason for entering debug mode.

There are two ways of accessing OSTAT content, as follows:

1. Send an `rstatus` command to the OnCE controller through the JTAG, or read the ONCE_STAT register through the AP access. Executing the `rstatus` command through the JTAG can be performed in both user and debug modes.
2. Perform an SDMA read access to the location in the SDMA core memory map (OSTAT register) debug mode using the `exec_once` command. With this method of access, the SDMA state reflected by the PST (processor status bit) is always DATA.

The register may also be accessed by a running application.

52.19.5 JTAG Interface Requirements

Because the signals received from the JTAG (running on TCK) are transferred to the OnCE controller (running on the SDMA clock), a synchronization mechanism is required.

52.19.5.1 TCK Speed Limitation

In the JTAG top-level layer, the TDO signal is always captured on a TCK falling edge. To guarantee a stable TDO signal from the SDMA during this operation, a falling edge detection is performed on TCK.

Before being latched in the *I* flip-flop (see Figure 52-75) on TCK falling edge, the TDO signal must be stable at the input of the flip-flop. This condition is verified if the TCK period is superior to the following delay:

worst-case edge detection delay + negative-edge signal propagation delay + JTAG top-level logic propagation delay

The frequency relationship, TCK < CLK/8, limitation guarantees that all operations are performed as expected.

52.19.5.2 Synchronization Implementation

Figure 52-75 shows the synchronization mechanism. Flip-flops tck0, tck1, and tck2 perform falling- and rising-edge detections on TCK. They generate the posedge_detected and negedge_detected nets that are used to sample the TDI and TMS inputs into the respective tdi and tms flip-flops, and update the tdo flip-flop to `yield` the TDO output. In the design, the only signal that might go metastable is the output of

the tck0 flip-flop. This signal is captured in the tck1 flip-flop and no logical operation is performed on it to minimize a metastability propagation risk.

The TDI and TMS flip-flops also cannot go metastable: The propagation time of the rising-edge detection signal through tck0, tck1, and tck2 guarantees that the TDI and TMS inputs are stable when captured in the TDI and TMS flip-flops.

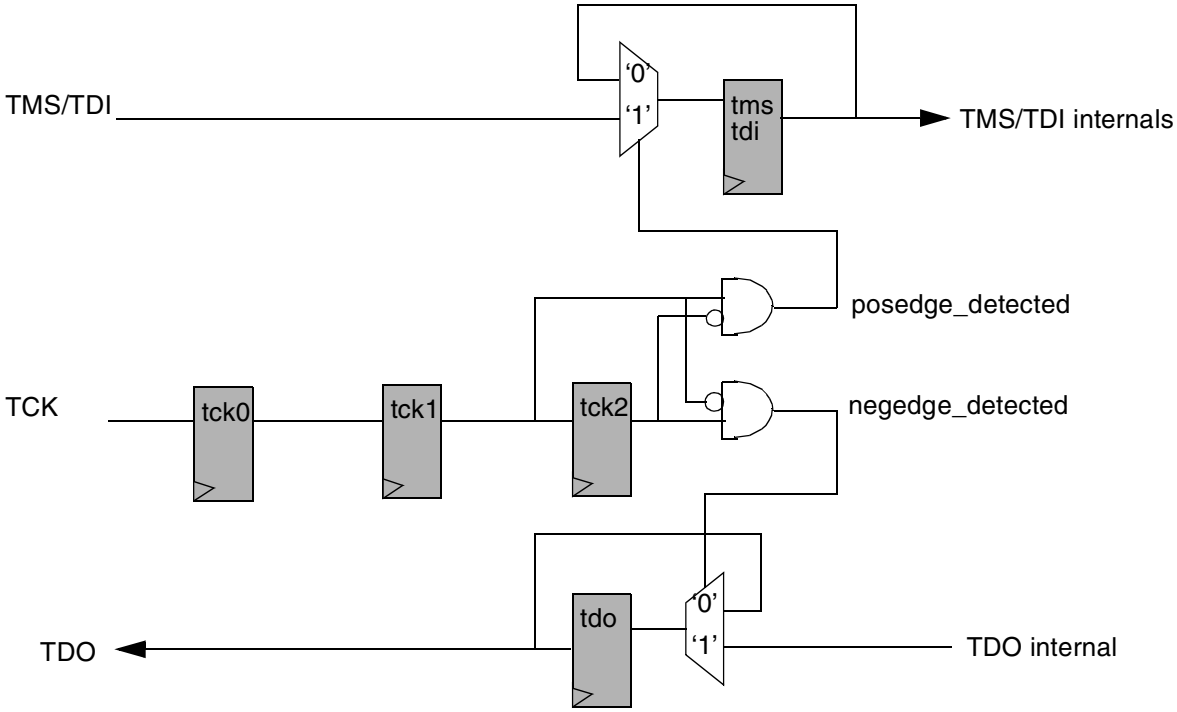


Figure 52-75. OnCE Synchronization Layer

Figure 52-76 shows synchronization timings. It takes three CLK clock cycles to synchronize TDI on the SDMA clock.

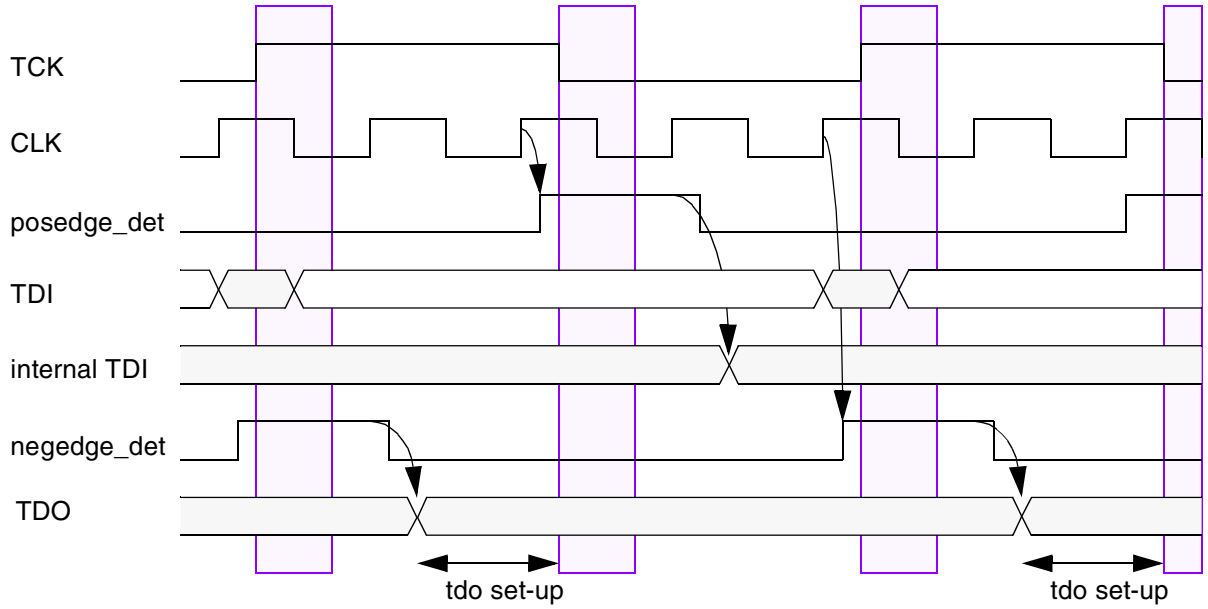


Figure 52-76. Synchronization Timings

52.19.5.3 JTAG Controller Start-Up Recommended Procedure

To ensure correct TAP controller initialization, it is recommended to use the following procedure:

1. Assert JTAG reset TRSTB (for example, set low).
2. Set TMS low.
3. Wait for 1 TCK clock.
4. Release JTAG reset TRSTB (for example, set high).
5. Wait for a minimum of five TCK cycles.

52.20 Using the OnCE

This section provides the elements necessary to run the OnCE module during a debug process. In addition to the basic set of commands described in Section 52.19.1, OnCE Commands," more complex commands can be built to meet users' requirements.

52.20.1 Activating Clocks in Debug Mode

For power consumption issues, some clocks in the SDMA are disabled when not needed. This is the case for instance when the SDMA is in sleep mode. Clock gating management depends on the interface used to control the OnCE.

- For the JTAG access, the SDMA clock gating must be turned off via the clk_gating_off input.
- For the AP access, the SDMA clock gating is automatically turned off when the AP access is enabled (see Section 52.12.3.16, OnCE Enable (ONCE_ENB)").

52.20.2 Getting the Current Status

Most of the commands the OnCE supports have an impact on the status of the SDMA. It is not permissible to request the execution of an instruction on the SDMA from the OnCE while the SDMA is not in debug mode. Such a violation may cause unpredictable behavior, and it might be necessary to reset the SDMA. Therefore, the value of the PST bits provided in the OnCE status register should always be checked before sending any request to the SDMA.

52.20.3 Methods of Entering Debug Mode

A debug request may be asserted at any time, but it is not always taken into account immediately. Debug mode cannot be entered in the middle of an instruction, or during the save or restore states of a context switch. The request is ignored when the core is already in debug mode. Refer to Figure 52-4, which shows all possible transitions to the debug state, as there are several ways to enter debug mode.

52.20.3.1 External Debug Request During Reset

To enter debug mode after exiting reset, the external debug line has to be maintained high. This line is handled by the JTAG top-level block.

NOTE

The SDMA detects the debug requests only if the SDMA clock is running (see Section 52.20.1, Activating Clocks in Debug Mode"). The debug request line should be not be maintained high when the SDMA is in debug mode.

The `debug_rqst` command (from the OnCE command set) is not supported during system reset.

52.20.3.2 Debug Request During Normal Activity

During normal activity, the SDMA enters debug mode when the following is true:

1. If the debug request line from the JTAG top-level is asserted, or
2. If the OnCE controller receives a debug_rqst command.

   The `debug_rqst` command can be sent by the JTAG access or by an access on the AP side (if the AP access is enabled).

52.20.3.3 Software Breakpoint Instruction

The SDMA enters debug mode at the end of the execution of a software breakpoint instruction. This instruction must be inserted in program flow executed by the core.

52.20.3.4 Event Detection Unit Matching Condition

If the event detection is enabled, a debug request is sent to the core after detecting a matching condition on the SDMA memory bus. See Section 52.20.6, OnCE Event Detection Unit" for more details.

52.20.4 Executing Instructions in Debug Mode

The OnCE supports a mechanism to execute instructions in debug mode. If the SDMA is in debug mode, then the `exec_once` command can be used to execute an SDMA instruction from the OnCE controller. The SDMA returns to debug mode at the end of each execution. Some instructions are not supported by the `exec_once` command: `done/yield/yiedge` (except `done` 5), BF, BT, BSF, BDF, JMP, JSR, JMPR, JSRR, RET, and LOOP, as well as all the illegal instructions are not supported.

NOTE

While instructions are executed in debug mode from the OnCE, the program counter of the SDMA is not incremented.

52.20.5 Command Sequences Examples

This section provides examples of command sequences that run the SDMA in debug mode. These sequences are available for both the AP and JTAG accesses.

The following presents the syntax used in this section. The data field provided with each command is put in parenthesis with the command name. A '-' is used if the data field provided is a *don't care* value.

```
my_command(data_field);    // executing my_command with a data field
my_command(-);             // executing my_command with a don't care data field
```

The value returned by the command (if there is one) is referred by an assignment. In case the value returned by the command is not used, the assignment is omitted. For an AP access, the value returned (it is always a data value) is obtained by reading back into the SDMA data register.

```
data_out = my_command(data_in);// returning a data value
```

To clarify the syntax, the instructions' opcodes are referred to by their names. In practice, use the corresponding 16-bit encoding.

52.20.5.1 Getting the SDMA Status

NOTE

Before executing any command that affects the SDMA (like `dmov` or `exec_once`), check that the SDMA is in debug mode.

Use the following snippet:

```
rstatus(); // read SDMA status until the SDMA is in debug mode
...
rstatus();
```

If the SDMA is not in debug mode, then a debug request must be generated. In this case, the SDMA enters debug mode at the end of the execution of the current instruction. Use this snippet:

```
debug_rqst(-); // debug request
```

In the following sections, it is assumed that the SDMA was successfully put into debug mode.

52.20.5.2 Saving the Context

The first debug task is to save the SDMA context, which is the content of the eight general-purpose registers, the loop and PC-related registers, and the flags. Use the general register GReg[1] as an intermediate register to export the entire context of the SDMA.

Example 52-5 shows how to save GReg[0], GReg[1], GReg[2] and GReg[3]. The sequence of commands used to export additional general registers is very similar to this.

Example 52-5. Save GReg[0], GReg[1], GReg[2], and GReg[3]

```
GReg1_data = dmov(-);                // the value exported is the content of GReg[1]
exec_once(“mov GReg1,GReg0”);        // puts the content of GReg[0] into GReg[1]
GReg0_data = dmov(-);                // the value exported is the content of GReg[0]
exec_once(“mov GReg1, GReg2”);       // puts the content of GReg[2] into GReg[1]
GReg2_data = dmov(-);                // the value exported is the content of GReg[2]
exec_once(“mov GReg1, GReg3”);       // puts the content of GReg[3] into GReg[1]
GReg3_data = dmov(-);                // the value exported is the content of GReg[3]
```

Get the value of the internal flags (SF, DF, T, and LM), of the loop related registers (EPC and SPC), and of the PC-related registers (PC and RPC). Use a `done` 5, which is the formatting instruction dedicated to the debug. This instruction formats the flags and the values contained in the registers. It also writes the resulting values into the channel context memory. It should not be used when entering debug from the IDLE state (for example, with no active channel script running on the SDMA), because it will update a channel context that may belong to any channel.

```
exec_once(“done 5”);       // formatting the value of flags and registers
```

At this point, the channel context should be up-to-date in memory, and debug operations should now be possible. However, the context can be exported with the following instructions:

Example 52-6. Exporting the Context

```
dmov(ctx_base_addr);                 // loading GReg[1] with the channel context base address
exec_once(“ld GReg0,(GReg1,0)”);     // get RPC-PC into GReg0
exec_once(“ld GReg1, (GReg1,1)”);    // get SPC-EPC into GReg1
Loop_data = dmov(-);                 // read back the value of Loop registers
exec_once(“mov GReg1, GReg0”);       // puts the PC info into GReg1
PC_data = dmov(-);                   // reads back the content of the PC registers
```

After this sequence of operations, the entire SDMA context is exported via the OnCE.

52.20.5.3 Restoring the Context

At this point in the operation, restore the context of the SDMA. It can be different from the original context located in memory, and the content previously saved into the debugging application via the OnCE. Example 52-7 shows how it is possible to modify the current channel context:

Example 52-7. Modifying the Current Channel Context

```
dmov(Loop_data);                    // put Loop former value into GReg[1]
exec_once(“mov GReg0, GReg1”);      // copy to GReg[0]
dmov(PC_data);                      // put PC former value into GReg[1]
exec_once(“mov GReg2, GReg1”);      // copy to GReg[2]
dmov(ctx_base_addr);                // put channel context base address into GReg[1]
exec_once(“st GReg0, (GReg1,1)”);   // restore Loop context
exec_once(“st GReg2, (GReg1,0)”);   // restore PC context
```

Once the context in memory is the desired context (with or without applying the previous instruction sequence), it can be restored to the *real* PC and loop registers in the SDMA core:

```
exec_once(“cpShReg”);      // restore flags and PC & loop related registers
```

After this command, the SDMA core PC, RPC, SPC, EPC registers, as well as the flags contain the same data as what is stored in the context RAM for the current channel.

Example 52-8 shows how to restore the context of general registers GReg[0], GReg[1], GReg[2] and GReg[3].

Example 52-8. Restoring the General Register Context

```
dmov(GReg3_data);          // put GReg[3] restore value in GReg[1]
exec_once(“mov GReg3, GReg1”);// restore GReg[3]
dmov(GReg2_data);          // put GReg[2] restore value in GReg[1]
exec_once(“mov GReg2, GReg1”);// restore GReg[2]
dmov(GReg0_data);          // put GReg[0] restore value in GReg[1]
exec_once(“mov GReg0, GReg1”);// restore GReg[0]
dmov(GReg1_data);          // restore GReg[1]
```

At this point, it is possible to restart the normal program execution.

NOTE

Every SDMA core general register value can be modified by a `mov` instruction, which makes modification of these registers easy during debug. Unfortunately, there is no such instruction as a `mov` to directly modify the contents of either PCU register or flag (PC, RPC, SPC, EPC, T, LM, SF, or DF). The `cpShReg` instruction is meant to provide a means for changing these register contents via the context memory.

52.20.5.4 Accessing the Memory

In the following example, it is assumed that the SDMA context is entirely saved. If true, it is permissible to modify the general purpose registers during debugging activity.

To perform a memory read access, the target address is stored via the OnCE in GReg[1], then the load instruction is executed on the SDMA (the data loaded from the memory overwrites the address contained in GReg[1]), and then the result value is read back via the OnCE.

```
macro READ:       dmov(target_addr);                 // put the target address in GReg[1]
                  exec_once(“ld GReg1,(GReg1,0)”);   // execute the load instruction
                  res_data = dmov(-);                // exports the result data value
```

For a memory write access, the target address is written in GReg[0], and the value to store is written in GReg[1]. Then the store instruction is executed on the SDMA.

```
macro WRITE:      dmov(target_addr);                 // puts the target address in GReg[1]
                  exec_once(“mov GReg0,GReg1”);      // puts the target address in GReg[0]
                  dmov(target_data);                 // puts the target data in GReg[1]
                  exec_once(“st GReg1,(GReg0,0)”);   // performs the store operation
```

This sequence is shown as an example; however, many other sequences are possible.

NOTE

This sequence of commands can also be applied to memory-mapped registers.

52.20.5.5 Resuming Program Execution

Before resuming program execution, it is assumed that the SDMA context is properly restored. There are two ways to restart the SDMA. Start by executing the last instruction fetched before entering debug mode, as follows.

```
run_core(-);              // resume execution from where we stopped before
```

If necessary, restart the execution from a different address. In this case, use the `exec_core` command. The data field provided with this command must be the encoding of a `jump` instruction.

```
exec_core(“jmp start_addr”);// rerun the SDMA from another address
```

In these two examples, the SDMA exits debug mode and keeps executing the code fetched from the memory.

52.20.5.6 Single Stepping in RAM

To execute a program step-by-step from the RAM, insert software breakpoints in the program flow at appropriate places so that the SDMA only executes one instruction before returning to debug mode.

First, read the next instruction to execute in the RAM. Then, depending on the value of this instruction, compute the address where a software breakpoint instruction should be inserted. The instruction at the corresponding address must be saved, and, the software breakpoint instruction is inserted. After restarting the SDMA, there is only one instruction executed before meeting the software breakpoint.

Example 52-9 shows the macro functions READ and WRITE, which correspond to the sequence of commands (described above) used to access the memory.

NOTE

The data read from the memory are 32-bit values, while the instructions are 16-bit values only. This is why it is best to only use addresses divided by two when accessing the memory.

Example 52-9. READ and WRITE Macro Functions

```
next_instr = READ(run_addr/2);      // read the next instruction to execute
// the tool now has to compute the address where the breakpoint
// instruction should be inserted, this address is the “bkpt_addr”
instr_save = READ(bkpt_addr/2);     // save the instruction before overwriting
STORE(“bkpt instruction”,bkpt_addr/2);// store the bkpt instruction in memory
exec_core(“jmp run_addr”);          // rerun the SDMA
rstatus(-);                         // wait for the SDMA to enter debug mode
...
rstatus(-);
STORE(instr_save,bpkt_addr/2);      // restore the instruction overwritten
```

In case of branched conditional instructions, a breakpoint instruction should be written at the two possible target addresses.

52.20.5.7 Single Stepping in ROM

No single-step mechanism is supported in ROM. The program code can be loaded in the RAM, where the single-step mechanism can be executed.

52.20.6 OnCE Event Detection Unit

The event detection unit watches signals from the data memory bus (DMBUS), which the SDMA core uses to access its RAM, ROM, and memory mapped registers. A debug request is sent to the OnCE controller when user-defined conditions on address and/or data values are true. See Figure 52-77.

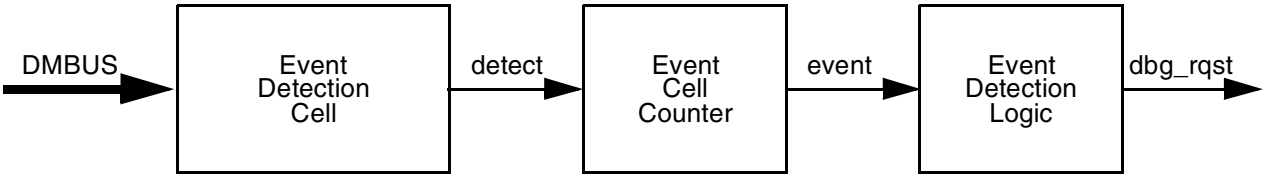


Figure 52-77. Event Detection Unit

A counter, provided with the detection cell, is decreased after an event detection. A debug request is sent to the core only when the counter reaches the value of 0. It is possible to disable the use of the counter if a debug request has to be generated after each event detection.

The event cell is the basic module that supports hardware breakpoints on an address value and/or data values coming from the SDMA memory bus. The trigger condition that generates the debug request is a mixed condition based on those values.

Figure 52-78 shows the event cell architecture. The event cell contains the address (stored in the memory address register) and the data (stored in the memory data register) used during the last memory access. There are some user-defined reference values located in memory mapped registers—the event cell addresses, the event cell address mask, the event cell data, and the event cell data mask. These registers are accessed by standard load/store instructions just like regular memory locations.

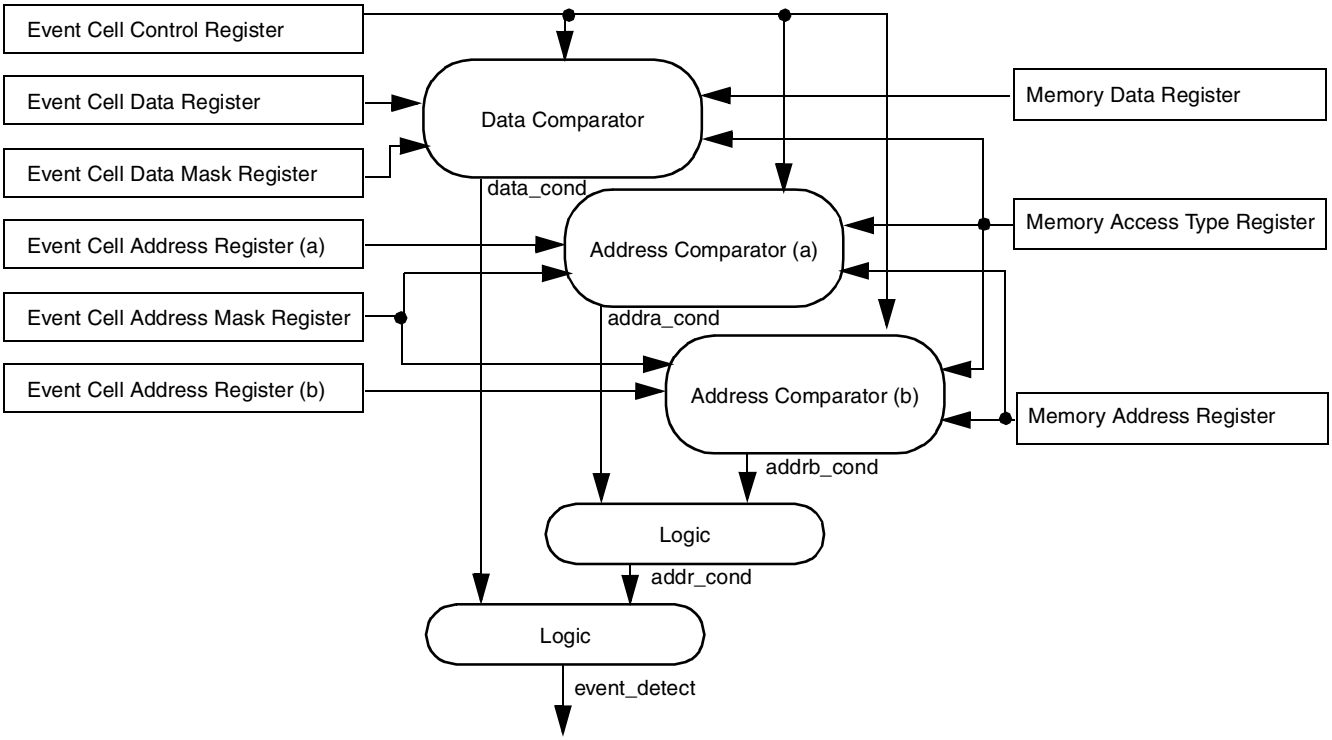


Figure 52-78. Event Cell Architecture

To define a memory breakpoint, three conditions are taken into account: The first two conditions are comparisons of the current memory address with user-defined reference addresses (these conditions are called addressA and addressB). The third condition consists of a comparison between the data received on the DMBUS and a user-defined reference data (this condition is called data). An intermediate address condition is set to express a dependency between addressA and addressB conditions.

52.20.7 Clock Gating and Reset

This section details how to use the clocks and handle the reset signals.

52.20.7.1 Clocks

Because the SDMA uses clock gating to save power, it is necessary to disable the clock gating and force the clocks to be enabled when using the OnCE. When the OnCE is accessed through its JTAG interface, clock gating must be disabled outside the SDMA via a dedicated SDMA input port clk_gating_off. The reason why detection is not performed automatically by the SDMA internal hardware is that it would cost power to monitor activity on the JTAG interface.

This clk_gating_off input is controlled by a control bit in the System JTAG Controller. Refer to the System JTAG Controller chapter for more information.

When the OnCE is accessed through the AP Control interface, clock gating is automatically turned off. This is done when bit 0 of the ONCE_ENB register (see Section 52.12.3.16, OnCE Enable (ONCE_ENB)") is set. A write access to this register is possible even when the OnCE clock is not running. If the AP access is used, the bit in the ONCE_ENB register must be set before any attempt to access any other OnCE register.

52.20.7.2 Resets

The OnCE reset is different from the SDMA main reset. The OnCE reset is connected to the TRST_B pin.

Normally, activating the SDMA reset while keeping the OnCE reset inactive (when possible) enables you to reset the core without having to reprogram the OnCE.

52.20.8 Real Time Features

To rebuild the skeleton of a program execution, it is necessary to store the addresses of the program instructions where jumps are taken: A trace buffer is therefore provided. A real time buffer has also been added to receive data values written during a program execution. The content of this register may be exported through JTAG ports without stopping the core.

52.20.8.1 Trace Buffer

The Trace Buffer is a 32-stage buffer that contains appropriate information to identify the 32 last changes of flow detected during a program execution. Figure 52-79 shows an overview of the Trace Buffer.

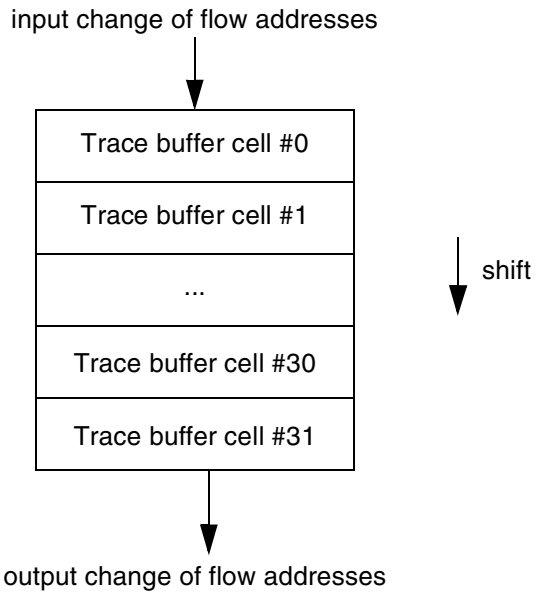


Figure 52-79. Trace Buffer

Each cell of the trace buffer contains two reference addresses and a flag. The flag is set when the addresses stored in the cell correspond to a valid change of flow; otherwise, the flag is cleared. The three most significant bits are unused.

After every change of flow detection, the address of current instruction and the address of the target instruction are stored at the top of the Trace Buffer (cell #0). The flag in the cell is set to indicate that a valid change of flow was detected. Former cell values are shifted one level down. The Trace Buffer contains the 32 last changes of flow. All the flags are reset on a software or a hardware reset, and after each transition from debug mode to user mode.

A memory mapped register of SDMA core, the Trace Buffer register (TB), is provided to read the content of the Trace Buffer. This operation should be done in debug mode. Performing a read access to the Trace Buffer register returns the content of the bottom of the Trace Buffer (cell #31). After every read access, the trace buffer is shifted one level down, and the flag at the top of the trace buffer is cleared.

A typical OnCE command sequence that retrieves the oldest change-of-flow information is a follows:

```
exec_once(“mov r1, TB”);          // stores the oldest change-of-flow in GReg1
dmov(-);                          // retrieves GReg1 contents
```

This sequence requires the SDMA to be put in debug mode.

This Register is also visible from AP side. Refer to section Section 52.12.3.26, OnCE Trace Buffer Register (OTB). The only difference is, while reading from AP side it can be done even when it is not in debug mode. But the ONCE ENB bit of Register OnCE Enable (ONCE_ENB) should be set.

52.20.8.2 Real Time Buffer

The Real Time Buffer register (RTB) is a memory mapped register that can be accessed as a regular memory location by the SDMA core during program execution. This register is located in the OnCE. Executing ar `rbuffer` command (see Section 52.19, The OnCE Controller” for further details) exports the content of this register through JTAG ports.

When a write access is performed at the memory location corresponding to the RTB, the receive flag (for example, the RCV bit) is set in the OnCE Status Register (OSR). This flag is cleared at the end of the execution of a `rbuffer` command.

NOTE

Every write access to the RTB memory location updates the RTB register even if the RCV flag is set. The RTB is cleared on a JTAG reset.

52.20.8.3 Emulation Pin

The debug_matched_event emulation pin reflects the matching condition status detected by the Event Detection Unit. Since it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

52.20.8.4 Real-Time Debug Outputs

Table 52-94 shows the debug signals that are available at the SDMA boundaries. Their availability at chip boundaries depends on the project.

Table 52-94. Real-Time Debug Output Pins

Pin	Description
debug_core_state[3:0]	The core_state bits reflect the state of the SDMA core. • The “Program” state is the usual instruction execution cycle. • The “Data” state is inserted when there are wait-states during a load or a store on the data bus (ld or st). • The “Change of Flow” state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions). • The “Change of Flow in Loop” state is used when an error causes a hardware loop exit. • The “Debug” state means the SDMA is in debug mode. • The “Functional Unit” state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf). • In “Sleep” modes, no script is running (this is the core idle state); the “after Reset” is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 is executed (boot operation). • The “in Sleep” states are the same as above except they do not have any corresponding channel: they are used when entering debug mode after reset; the reason is that it is necessary to return to the “Sleep after Reset” state when leaving debug mode. 0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Context Switch Saving Channel 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change of Flow in Loop in Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Context Switch Restoring Channel
debug_yield	Pulse that is active when a `yield` (`done` 0) or a `yieldge` (`done` 1) instruction is executed. 0 - 1 `yield`/`yieldge` executed
debug_core_run	Active when the SDMA core is executing instructions. 0 Debug or sleep mode 1 Run mode
debug_event_channel_sel	Indicates if debug_event_channel displays current channel or last received event 0- debug_event_channel[5:0] gives the number of the current channel 1- debug_event_channel[5:0] gives the number of the last received event

Table 52-94. Real-Time Debug Output Pins (continued)

Pin	Description
debug_event_channel[5:0]	Gives the number of any DMA request as soon as it is received or the number of the current channel. The value of debug_event_channel_sel indicates if debug_event_channel displays the current channel or last received event. The signal debug_event_channel_sel must be observed to determine what information is provided on debug_event_chanel at any given time.
debug_pc[13:0]	Program Counter value; it has a meaning when the core is in run mode.
debug_mode	Set when the core is in debug. 0 - 1 Core is in debug
debug_bus_error	Set when an error was received during a load or a store (`ld`, `st`, `ldf`, or `stf` instruction) and registered in SF or DF flag. 0 No error during last load/store 1 Error during last load/store
debug_bus_device[4:0]	Indicates the device or functional unit that is accessed by the current instruction. The debug_bus_device output is always valid when in sleep mode, debug mode, or executing any instruction that does not access the functional units or the memory mapped devices, “no access” is output. 0 No access 1 MSA 2 MDA 3 MD 4 MS 5 Reserved 6 Reserved 7 Reserved 8 Reserved 9 DSA 10 DDA 11 DD 12 DS 13 CA 14 CS 15 Reserved 16 Memory (RAM or ROM) 17 Memory mapped register 18 Peripheral #1 19 Peripheral #2 20 Peripheral #3 21 Peripheral #4 22 Peripheral #5 23 Peripheral #6 24 Peripheral #7 25 Peripheral #8 26 Peripheral #9 27 Peripheral #10 28 Peripheral #11 29 Peripheral #12 30 Peripheral #13 31 Peripheral #14

Table 52-94. Real-Time Debug Output Pins (continued)

Pin	Description
debug_bus_rwb	Indicates the direction of the access given by debug_bus_device 0 Write access (st or stf) 1 Read access (ld or ldf)
debug_matched_dmbus	Pulse indicating the OnCE event detection unit has detected a match on the data bus during an access to memory (RAM or ROM), a memory mapped register or a peripheral that is hooked to the SDMA. 0 - 1 data bus match detected
debug_rtbuffer_write	Pulse indicating when the real-time buffer is written by the core. 0 - 1 RTB was modified
debug_evt_chn_lines[7:0]	Eight lines that generate short pulses when DMA requests are received or channels are (re)started. Every line is controlled through two parameters defined in registers Cross-Trigger Events Configuration Register (1) and (2) (XTRIG_CONF1 and XTRIG_CONF2) (as described in Section 52.12, AP Memory Map and Control Register Definitions,"). The following two parameters are available for every line: • CNF—Indicates what is monitored on the line: 0 for a channel start, 1 for a DMA request reception • NUM[5:0]—Gives the number of the DMA request or channel to monitor

The matched_event emulation pin reflects the matching condition status detected by the Event Detection Unit. Because it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

All real-time debug outputs are disabled by default (for example, they are stuck to 0) to avoid power consumption when they are not used. They are enabled when bit 11 (RTDOBS) of the Configuration Register (CONFIG) is set. Signals provided to the system JTAG controller for SDMA debug mode status will also be enabled when the *clk_gating_off* input is asserted.

52.21 Instruction Set

52.21.1 Instruction Encoding

This section presents a short summary of the instruction codes. All context switch instructions are listed for information only; they cannot function properly out of the context switch routine.

```
x...x           - don't care
rrr             - destination/source general register
sss             - additional source general register
bbb             - general register used as address base register
ddddd           - address displacement
nnnnn           - bit number
uuuuuuuu        - function unit command bits
pppppppp        - branch displacement (signed)
iiiiiiii        - 8-bit immediate
jjj             - control bit to clear
ff              - flag to clear
```

```
00000jjj00000000  - done (done,yield,wait)
00000jjj00000001  - notify
00000xxx00000010  - reserved
00000xxx00000011  - reserved
00000xxx00000100  - reserved
0000000000000101  - softBkpt
0000000100000101  - reserved
0000001000000101  - reserved
0000001100000101  - reserved
0000010000000101  - reserved
0000010100000101  - reserved
0000011000000101  - reserved
0000011100000101  - reserved
0000000000000110  - ret
0000000100000110  - reserved
0000001000000110  - reserved
0000001100000110  - reserved
0000010000000110  - reserved
0000010100000110  - reserved
0000011000000110  - reserved
0000011100000110  - reserved
000000ff00000111  - clrf ff
0000010000000111  - reserved
0000010100000111  - reserved
0000011000000111  - reserved
0000011100000111  - illegal
00000rrr00001000  - jmpr r
00000rrr00001001  - jsrr
00000rrr00001010  - ldrpc r
00000rrr00001011  - reserved
00000rrr000011xx  - reserved
00000rrr00010000  - revb
00000rrr00010001  - revblo
00000rrr00010010  - rorb
00000rrr00010011  - reserved
00000rrr00010100  - ror1
00000rrr00010101  - lsr1
00000rrr00010110  - asr1
00000rrr00010111  - lsl1
00000rrr001nnnnn  - bclri r,n
00000rrr010nnnnn  - bseti r,n
00000rrr011nnnnn  - btsti r,n
00000xxx10000xxx  - reserved
00000rrr10001sss  - mov
00000rrr10010sss  - xor
00000rrr10011sss  - add
00000rrr10100sss  - sub
00000rrr10101sss  - or
00000rrr10110sss  - andn
00000rrr10111sss  - and
00000rrr11000sss  - tst
00000rrr11001sss  - cmpeq
00000rrr11010sss  - cmplt
00000rrr11011sss  - cmphs
0000011011100000  - reserved
0000011011100001  - reserved
```

```
0000011011100010  - cpShReg
0000011011100011  - reserved
0000011011100100  - reserved
0000011011100101  - reserved
0000011011100110  - reserved
0000011011100111  - reserved
00000xxx11101xxx  - reserved
00000xxx11110xxx  - reserved
00000xxx11111xxx  - reserved
00001rrriiiiiiii  - ldi r,i
00010rrriiiiiiii  - xori r,i
00011rrriiiiiiii  - addi r,i
00100rrriiiiiiii  - subi r,i
00101rrriiiiiiii  - ori r,i
00110rrriiiiiiii  - andni r,i
00111rrriiiiiiii  - andi r,i
01000rrriiiiiiii  - tsti r,i
01001rrriiiiiiii  - cmpeqi r,i
01010rrrdddddbbb  - ld r,(d,b)
01011rrrdddddbbb  - st r,u
01100rrruuuuuuuu  - ldf r,u
01101rrruuuuuuuu  - stf r,u
011100xxxxxxxxxx  - reserved
011101xxxxxxxxxx  - reserved
011110ffnnnnnnnn  - Loop ff flags are reset
01111100pppppppp  - bf pc=pc+signed(pppppppp)+1
01111101pppppppp  - bt pc=pc+signed(pppppppp)+1
01111110pppppppp  - bsf pc=pc+signed(pppppppp)+1
01111111pppppppp  - bdf pc=pc+signed(pppppppp)+1
10aaaaaaaaaaaaaa  - jmp absolute
11aaaaaaaaaaaaaa  - jsr absolute
```

52.21.2 SDMA Instruction Set

This section describes all the useful instructions from the SDMA set. Table 52-95 summarizes the instructions.

Table 52-95. SDMA Instruction List

Instruction	Description	Page
ADD	Addition	on page 52-170
ADDI	Add with Immediate Value	on page 52-171
AND	Logical AND	on page 52-172
ANDI	Logical AND with Immediate Value	on page 52-173
ANDN	Logical AND NOT	on page 52-174
ANDNI	Logical AND with Negated Immediate Value	on page 52-175
ASR1	Arithmetic Shift Right by 1 Bit	on page 52-176
BCLRI	Bit Clear Immediate	on page 52-177
BDF	Conditional Branch if Destination Fault	on page 52-178
BF	Conditional Branch if False	on page 52-179

Table 52-95. SDMA Instruction List (continued)

Instruction	Description	Page
BSETI	Bit Set Immediate	on page 52-180
BSF	Conditional Branch if Source Fault	on page 52-181
BT	Conditional Branch if True	on page 52-182
BTSTI	Bit Test immediate	on page 52-183
CLRF	Clear CPU flags	on page 52-184
CMPEQ	Compare for Equal	on page 52-185
CMPEQI	Compare with Immediate for Equal	on page 52-186
CMPHS	Compare for Higher or Same	on page 52-187
CMPLT	Compare for Less Than	on page 52-188
cpShReg	Update Context of PCU Registers and Flags	on page 52-189
DONE	DONE, Yield	on page 52-190
ILLEGAL	ILLEGAL Instruction	on page 52-192
JMP	Unconditional Jump Immediate	on page 52-193
JMPR	Unconditional Jump	on page 52-194
JSR	Unconditional Jump to Subroutine Immediate	on page 52-195
JSRR	Unconditional Jump to Subroutine	on page 52-196
LD	Load Register	on page 52-197
LDF	Load Register from Functional Unit	on page 52-199
LDI	Load Register with Immediate Value	on page 52-201
LDRPC	Load from RPC to Register	on page 52-202
LOOP	Hardware Loop	on page 52-203
LSL1	Logical Shift Left by 1 Bit	on page 52-206
LSR1	Logical Shift Right by 1 Bit	on page 52-207
MOV	Logical Move	on page 52-208
NOTIFY	Notify to MCU	on page 52-209
OR	Logical OR	on page 52-210
ORI	Logical OR with Immediate Value	on page 52-211
RET	Return from Subroutine	on page 52-212
REVB	Reverse Byte Order	on page 52-213
REVBLO	Reverse Low Order Bytes	on page 52-214
ROR1	Rotate Right by 1 Bit	on page 52-215
RORB	Rotate Right by 1 Byte	on page 52-216
SOFTBKPT	Software Breakpoint	on page 52-217

Table 52-95. SDMA Instruction List (continued)

Instruction	Description	Page
ST	Store Register	on page 52-218
STF	Store Register in Functional Unit	on page 52-219
SUB	Subtract	on page 52-222
SUBI	Subtract with Immediate	on page 52-223
TST	Test with Zero	on page 52-224
TSTI	Test Immediate	on page 52-225
XOR	Logical Exclusive OR	on page 52-226
XORI	Exclusive OR with Immediate	on page 52-227

ADD

Addition

Operation:

```
GReg[r] ← GReg[s] + GReg[r]
T ← (GReg[r] == 0)
```

Assembler:

Syntax: `add r,s`

Example: `add 0,3`
to ADD GReg[3] and GReg[0] and store the result in GReg[0]

CPU Flags: T

Cycles: 1

Description: Performs the ADDition of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*. The T flag is set if the result of the operation is 0. It is cleared if the result is not 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	1	s	s	s

Instruction Fields:

`rrr` - destination register field:

```
000 - GReg[0]
001 - GReg[1]
010 - GReg[2]
011 - GReg[3]
100 - GReg[4]
101 - GReg[5]
110 - GReg[6]
111 - GReg[7]
```

`sss` - source register field:

```
000 - GReg[0]
001 - GReg[1]
010 - GReg[2]
011 - GReg[3]
100 - GReg[4]
101 - GReg[5]
110 - GReg[6]
111 - GReg[7]
```

ADDI

Add with Immediate Value

Operation:

```
GReg[r] ← GReg[r] + immediate
T ← (GReg[r] == 0)
```

Assembler:

Syntax: `addi r,immediate`

Example: `addi 6,112`
to ADD GReg[6] and decimal value 112 and store the result in GReg[6]

CPU Flags: T

Cycles: 1

Description: Add a 0-extended immediate value to a general register; stores the result in the general register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

AND

Logical AND

Operation:

GReg[r] ← GReg[s] & GReg[r]

Assembler:

Syntax: `and r,s`

Example: `and 1,2`
to AND GReg[1] and GReg[2] and store the result in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	1	1	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

ANDI

Logical AND with Immediate Value

Operation:

GReg[r] ← GReg[r] & immediate

Assembler:

Syntax: `andi r,immediate`

Example: `andi 7,45`
to AND GReg[7] and decimal value 45 and store the result in GReg[7]

CPU Flags: unaffected

Cycles: 1

Description: Performs an AND between a 0-extended immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

ANDN

Logical AND NOT

Operation:

GReg[r] ← ~GReg[s] & GReg[r]

Assembler:

Syntax: `andn r,s`

Example: `andn 3,4`
to AND GReg[3] and NOT GReg[4] (bit inverted) and store the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the negation of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	1	0	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

ANDNI Logical AND with Negated Immediate Value

Operation:

GReg[r] ← GReg[r] & ~immediate

Assembler:

Syntax: andni r,immediate

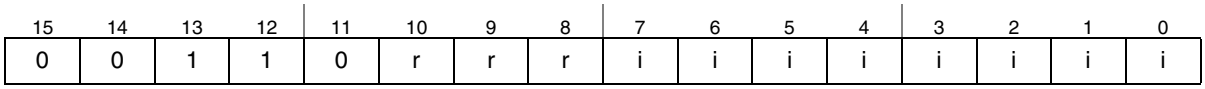
Example: andni 0,2
to AND GReg[0] and decimal value -3 (inverted 32-bit value 2) and store the result in GReg[0]

CPU Flags: unaffected

Cycles: 1

Description: Performs an AND between the negation of a 0-extended 8-bit immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:
000 - GReg[0]
001 - GReg[1]
010 - GReg[2]
011 - GReg[3]
100 - GReg[4]
101 - GReg[5]
110 - GReg[6]
111 - GReg[7]

iiiiiiii - immediate value:
00000000 - 0
00000001 - 1
...
11111110 - 254
11111111 - 255

ASR1 Arithmetic Shift Right by 1 Bit

Operation:

`GReg[r]:{b31,b30,...,b1,b0} ← GReg[r]:{b31,b31,b30,...,b1}`

Assembler:

`Syntax: asr1 r`

Example: asr1 3
to divide by 2 the signed value of GReg[3] and store the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any general register to the right and keep the same sign: The left bit (bit 31) is kept untouched.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	0

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

BCLRI

Bit Clear Immediate

Operation:

```
GReg[r]:{b31,...,b(i+1),0,b(i-1),...,b0} ←
GReg[r]:{b31,...,b(i+1),b(i),b(i-1),...,b0}
```

Assembler:

Syntax: `bclri r,i`

Example: `bclri 1,12`
to clear bit 12 in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Clear the bit of register r specified by the 5-bit immediate field

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	1	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiii` - bit number field:
`00000` - bit 0
`00001` - bit 1
...
`11110` - bit 30
`11111` - bit 31

BDF

Conditional Branch if Destination Fault

Operation:

`if (DF == 1) PC ← PC + 1 + displacement else PC ← PC + 1`

Assembler:

Syntax: `bdf label`

Example: bdf LLL
To jump to LLL if DF is set, or go to the next instruction if DF is cleared; the displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag DF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag DF is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	1	p	p	p	p	p	p	p	p

Instruction Fields:

`pppppppp` - signed displacement field:
`00000000` - 0
`00000001` - 1
...
`01111110` - 126
`01111111` - 127
`10000000` - -128
`10000001` - -127
...
`11111110` - -2
`11111111` - -1

BF

Conditional Branch if False

Operation:

```
if (T == 0) PC ← PC + 1 + displacement
else PC ← PC + 1
```

Assembler:

Syntax: `bf label`

Example: `bf LLL`
To jump to LLL if T is cleared, or go to the next instruction if T is set. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is cleared, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is set, no jump is performed: The next instruction is located at the next PC address.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	0	p	p	p	p	p	p	p	p

Instruction Fields:

`pppppppp` - signed displacement field:
`00000000` - 0
`00000001` - 1
...
`01111110` - 126
`01111111` - 127
`10000000` - -128
`10000001` - -127
...
`11111110` - -2
`11111111` - -1

BSETI

Bit Set Immediate

Operation:

```
GReg[r]:{b31,...,b(i+1),1,b(i-1),...,b0} ←
GReg[r]:{b31,...,b(i+1),b(i),b(i-1),...,b0}
```

Assembler:

Syntax: `bseti r,i`

Example: bseti 6,5
to set bit 5 in GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Sets bit number i in the selected General Register.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	0	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiii` - bit number field:
`00000` - bit 0
`00001` - bit 1
...
`11110` - bit 30
`11111` - bit 31

BSF

Conditional Branch if Source Fault

Operation:

```
if (SF == 1) PC ← PC + 1 + displacement
else PC ← PC + 1
```

Assembler:

Syntax: `bsf label`

Example: `bsf LLL`
To jump to LLL if SF is set, or go to the next instruction if SF is cleared. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag SF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag SF is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	0	p	p	p	p	p	p	p	p

Instruction Fields:

`pppppppp` - signed displacement field:
`00000000` - 0
`00000001` - 1
...
`01111110` - 126
`01111111` - 127
`10000000` - -128
`10000001` - -127
...
`11111110` - -2
`11111111` - -1

BT

Conditional Branch if True

Operation:

```
if (T == 1) PC ← PC + 1 + displacement
else PC ← PC + 1
```

Assembler:

Syntax: `bt label`

Example: bt LLL
To jump to LLL if T is set, or go to the next instruction if T is cleared. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	1	p	p	p	p	p	p	p	p

Instruction Fields:

`pppppppp` - signed displacement field:
`00000000` - 0
`00000001` - 1
...
`01111110` - 126
`01111111` - 127
`10000000` - -128
`10000001` - -127
...
`11111110` - -2
`11111111` - -1

BTSTI

Bit Test immediate

Operation:

`T ← GReg[r]:b(i)`

Assembler:

Syntax: `btsti r,i`

Example: `btsti 2,29`
to test bit 29 in GReg[2] and copy its value in flag T

CPU Flags: T

Cycles: 1

Description: T is loaded with the value of bit number i from the selected general register.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	1	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iii` - bit number field:
`00000` - bit 0
`00001` - bit 1
...
`11110` - bit 30
`11111` - bit 31

CLRF

Clear CPU flags

Operation:

```
if (ff%2 == 0) SF ← 0
if (ff/2 == 0) DF ← 0
```

Assembler:

Syntax: `clrf ff`

Example: clrf 2
to clear flag SF and keep flag DF unchanged

CPU Flags: SF, DF

Cycles: 1

Description: Clears a selection of the CPU fault flags: SF, DF, both SF and DF or none can be cleared.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	f	f	0	0	0	0	0	1	1	1

Instruction Fields:

`ff` - flags field:
`00` - clear `SF` and clear `DF`
`01` - clear `DF`
`10` - clear `SF`
`11` - no clear

CMPEQ

Compare for Equal

Operation:

```
T ← (GReg[s] == GReg[r])
```

Assembler:

Syntax: `cmpeq r,s`

Example: `cmpeq 7,5`
to compare GReg[7] and GReg[5] and set flag T if they are equal

CPU Flags: T

Cycles: 1

Description: Subtracts the destination general register *r* from the source general register *s*, and sets T if the result is 0, clears T if the result is not 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	1	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`sss` - source register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

CMPEQI Compare with Immediate for Equal

Operation:

```
T ← (GReg[r] == immediate)
```

Assembler:

Syntax: `cmpeqi r,immediate`

Example: `cmpeqi 2,13`
To compare GReg[2] and decimal value 13 and set flag T if they are equal

CPU Flags: T

Cycles: 1

Description: Subtracts the 0-extended 8-bit immediate value from the general register, and sets T if the result is 0, clears T if the result is not 0. The immediate value is the low-order byte of the instruction.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

CMPHS

Compare for Higher or Same

Operation:

T ← (GReg[r] ≥ GReg[s])

Assembler:

Syntax: `cmphs r,s`

Example: `cmphs 0,1`
To compare GReg[0] and GReg[1] and set flag T if GReg[0] is higher than or equal to GReg[1]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is higher than or equal to the source general register *s*, clears T otherwise. The comparison is unsigned.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	1	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`sss` - source register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

CMPLT

Compare for Less Than

Operation:

T ← (GReg[r] < GReg[s])

Assembler:

Syntax: `cmplt r,s`

Example: `cmplt 7,4`
To compare GReg[7] and GReg[4] and set flag T if GReg[7] is lower than GReg[4]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is lower than the source general register *s*, clears T otherwise. The comparison is signed.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	0	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

cpShReg Update Context of PCU Registers and Flags

Assembler:

Syntax: cpShReg

CPU Flags: none

Cycles: 1

Description: SF, RPC, T, PC,LM, EPC, DF, and SPC registers are updated according to the value of their corresponding bits in the context memory. This instruction must only be used in debug mode via the OnCE. It reverses the done 5 operation.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	1	1	1	0	0	0	1	0

Instruction Fields:

DONE

DONE, Yield

Operation:

```
if (jjj&6 == 2) HE[CCR] ← 0
if (jjj == 3) HI[CCR] ← 1
if (jjj == 4) EP[CCR] ← 0
if ((jjj == 0) && (NCP > CCP)) CCR ← NCR
else if ((jjj == 1) && (NCP >= CCP)) CCR ← NCR
else CCR ← NCR
```

(CCR stands for Current Channel Register; NCR stands for Next Channel Register)

Assembler:

Syntax: `done jjj`

Example: `done 3`
To clear HE bit for the current channel, send an interrupt to the AP for the current channel and reschedule.

CPU Flags: Unaffected

Cycles: Variable if a context switch is done, 1 otherwise

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required. Sends an interrupt to the corresponding CPU (AP) by setting the appropriate flag, if required (HI for the corresponding channel number). Reschedules according to the mode and the NCP (Next Channel Priority) and CCP (Current Channel Priority) values. According to the scheduling decision, the NCR (Next Channel Register) is copied to the CCR (Current Channel Register) and channel contexts are switched.
If several channels with the same highest priority are pending, they are ordered by their number from 31 down to 0. The higher number is selected (for example, channel 26 is selected if channels 3, 12, 14, and 26 with the same highest priority are pending).
If no flag is modified, the reschedule can allow the replacement of the current channel by another channel with a priority strictly greater than the current channel priority (`yield`). Or, it can allow the replacement of the current channel by another channel with a priority greater than or equal to the current channel priority (`yieldge`). In the latter case, the selected channel will always be the first one with the same priority, starting from channel number 31 down to channel 0 (the current channel does not belong to the set of selectable channels).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	0

Instruction Fields:

jjj - Channel Flags field:
000 - No channel flags affected: Reschedule only if the next channel priority is greater than current channel priority (yield)
001 - No channel flags affected: Reschedule only if the next channel priority is greater than or equal to the current channel priority (yieldge)
010 - Clear HE for the current channel and reschedule
011 - Clear HE, set HI for the current channel and reschedule
100 - Clear EP for the current channel and reschedule
101 - Reserved for debug to copy relevant registers into context memory
110 - RESERVED
111 - RESERVED

For the scheduling rules, refer to Section 52.4.3, Scheduler Functional Description." Every possible done instruction is further described as follows:

- done 0/yield is executed by a channel script when it accepts preemption by a higher priority channel;
- done 1/yieldge is executed by a channel script when it accepts preemption by a higher priority channel and it also accepts a roll-up with other channels that have the same priority;
- done 2 is executed by a channel script that was triggered by a AP start via the Channel Start (HSTART) register, when its task is completed and it requires termination;
- done 3 is executed by a channel script that was triggered by a AP start via the Channel Start (HSTART) register, when its task is completed, it requires termination and it needs to trigger an interrupt to the AP upon closure;
- done 4 is executed by a channel script that was triggered by a DMA request, when its task is completed and it requires termination;
- done 5 is used in debug mode only; it copies the PCU registers and flags to the context memory of the current channel;

ILLEGAL

ILLEGAL Instruction

Operation:

PC ← 0001

Assembler:

Syntax: illegal

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the Illegal instruction routine located at address 0001. All unauthorized instructions result in an Illegal instruction behavior; however, the ILLEGAL instruction must be used to guarantee software compatibility with future versions of the SDMA.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Instruction Fields:

JMP

Unconditional Jump Immediate

Operation:

PC ← absolute_address

Assembler:

Syntax: `jmp label`

Example: `jmp LLL`
the assembler translates the label to the exact address

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	a	a	a	a	a	a	a	a	a	a	a	a	a	a

Instruction Fields:

`aaaaaaaaaaaaaa` - address field:
`00000000000000` - 0
`00000000000001` - 1
...
`11111111111110` - 16382
`11111111111111` - 16383

JMPR

Unconditional Jump

Operation:

PC ← GReg[r]

Assembler:

Syntax: `jmpr r`

Example: `jmpr 0`
To jump to address stored in GReg[0]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the absolute address contained in a General Register.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	0

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

JSR Unconditional Jump to Subroutine Immediate

Operation:

RPC ← PC + 1
PC ← absolute_address

Assembler:

Syntax: `jsr r`

Example: `jsr LLL`
Jumps to subroutine starting at LLL; the assembler translates the label to exact address

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine located at the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	a	a	a	a	a	a	a	a	a	a	a	a	a	a

Instruction Fields:

`aaaaaaaaaaaaaa` - address field:
`00000000000000` - 0
`00000000000001` - 1
...
`11111111111110` - 16382
`11111111111111` - 16383

JSRR

Unconditional Jump to Subroutine

Operation:

```
RPC ← PC + 1
PC ← GReg[r]
```

Assembler:

Syntax: `jsrr r`

Example: `jsrr 5`
Jumps to subroutine located at address stored in GReg[5]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine at address contained in a General Register

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	1

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

LD

Load Register

Operation:

```
GReg[r] ← [GReg[b] + displacement]
if (transfer_error) SF ← 1
else SF ← 0
```

Assembler:

Syntax: `ld r,(b,displacement)`

Example: `ld 1,(2,23)`
Loads data into GReg[1]; the data is located at address obtained by adding decimal value 23 to GReg[2]

CPU Flags: SF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to fetch on the DM bus. The data received from the bus is stored in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	r	r	r	d	d	d	d	d	b	b	b

Instruction Fields:

`rrr` - destination register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
...
`111` - `GReg[7]`

`bbb` - base address register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
...
`111` - `GReg[7]`

`ddddd` - displacement value:
`00000` - 0
`00001` - 1

...
11111 - 31

LDF

Load Register from Functional Unit

Operation:

```
GReg[r] ← [fu_address]
if (transfer_error) SF ← 1
else SF ← 0
```

fu_address is an 8-bit field and depends on addressed functional unit

Assembler:

Syntax: `ldf r,fu_address`

Example: `ldf 0,13`
Loads data coming from the Burst DMA register MD into GReg[0]; it is a 32-bit access with no prefetch

CPU Flags: SF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

Description: Sends an 8-bit address on the Functional Unit Bus (FU bus) and stores the data received from the bus in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the LDF instruction usage with each functional unit:

- Section 52.18.1.6, Burst DMA Read (ldf)" for Burst DMA
- Section 52.18.2.6, Burst DMA2 Read (ldf) for BP DMA

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`ffffffff` - functional unit source register and action (unspecified values are reserved):
`00000000` - MSA

00000100 - MDA
00001001 - MD byte
00001010 - MD halfword
00001011 - MD word
00001100 - MS
00101001 - MD byte - prefetch
00101010 - MD halfword - prefetch
00101011 - MD word - prefetch
01000000 - DSA
01000100 - DDA
01001001 - DD byte
01001010 - DD halfword
01001011 - DD word
01001100 - DS
01101001 - DD byte - prefetch
01101010 - DD halfword - prefetch
01101011 - DD word - prefetch
11000000 - PSA
11001000 - PD
11010000 - PDA
11011000 - PD in copy mode (rrr contents are lost)
11101000 - PD - prefetch next data
11111111 - PS

LDI

Load Register with Immediate Value

Operation:

GReg[r] ← immediate

Assembler:

Syntax: `ldi r,immediate`

Example: `ldi 6,1`
loads decimal value 1 into GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Stores a 0-extended immediate value in a General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

LDRPC

Load from RPC to Register

Operation:

GReg[r] ← RPC

Assembler:

Syntax: `ldrpc r`

Example: `ldrpc 3`
copies RPC to GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Stores the contents of the RPC in a General Register. That instruction may be used to have more than one level of subroutines.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	1	0

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

LOOP

Hardware Loop

Operation:

```
if (ff%2 == 0) SF ← 0
if (ff/2 == 0) DF ← 0
if ((GReg[0] == 0) || (SF == 1) || (DF == 1))
            PC ← PC + loop_size + 1
else
        {
            SPC ← PC + 1
            EPC ← PC + loop_size + 1
            LM ← 1
            PC ← PC + 1
        }
```

during every instruction execution in the loop:

```
if ((SF == 1) || (DF == 1))
        {
            LM ← 0
            PC ← EPC
        }
else if ((PC + 1) == EPC)
        {
            GReg[0] ← GReg[0] - 1
            if (GReg[0] == 0)
            {
                LM ← 0
                PC ← EPC
            }
            else PC ← SPC
        }
else PC ← nextPC(instruction)
```

after the execution of the last instruction of the loop body:

```
if (GReg[0] == 0)
            T ← 1
else
            T ← 0
```

Assembler:

Syntax: `loop n{,ff}`

Example: `loop 3,1`

Executes GReg[0] times the instructions comprised between PC+1 and PC+3 (included); ff=1 clears the DF flag before starting the loop. When omitted, the `ff` field is set to 0 (clearing both SF and DF).

CPU Flags: LM[1:0], T

Cycles: 2 when the loop count (GReg[0]) is 0 or SF or DF is set at loop start, 1+1 when the loop starts but exits abnormally (SF or DF set inside the loop which adds 1 cycle to the offending load or store to jump to EPC), 1 when the loop is executed normally

Description: The `loop` instruction executes a sequence of instructions several times. The number of times is given by the contents of GReg[0], the loop counter. SDMA will jump to the first instruction after the end of the loop if the value in GReg[0] is 0. Otherwise the SDMA enters loop mode. It sets the most significant bit of the LM flag that will only be reset once the last instruction of the last loop is executed. The instructions in the loop are executed GReg[0] times.
The management of fault flags (SF and DF) is as follows. When entering the hardware loop, SF and DF can be cleared according to the ff field of the instruction. After that operation, if any flag is still set the loop will not be executed. The SDMA will jump to the first instruction after the end of the loop without entering loop mode. During the execution of the loop, if any fault flag is set by a LD, LDF, ST, or STF instruction, the SDMA will immediately exit loop mode and jump to the first instruction after the end of the loop. In that case, GReg0 is not decremented for that last piece of the loop body execution (even if the SF or DF flag is set at the last instruction of the loop body).
The T flag reflects the state of GReg[0] after the end of the loop, which is an indicator of the complete execution of the loop. If the loop exited because of an error (SF or DF set), GReg[0] will not be 0 at the end of the loop, hence T will be cleared. If the loop executes without fault, GReg[0] will be 0 at the end of the loop, hence T will be set. The boundary case when a source or destination fault occurs at the last instruction of the last loop is considered as an anticipated exit of the loop, which causes the T flag to be cleared. If the last instruction executed before leaving the hardware loop also tries to modify the T flag, the flag is updated according to the value of GReg[0], NOT according to the result of the last executed instruction.

Limitations: 1. Jump instructions (JMP, JMPR, JSR, JSRR, BF, BT, BSF, BDF) are not allowed inside the hardware loop.
2. GReg[0] cannot be written to inside the hardware loop (it can be read).
3. The empty loop (0 instruction in the body) is forbidden.
4. If GReg[0] == 0 at the start of the loop, which causes a jump to EPC, the T flag is not updated.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	f	f	n	n	n	n	n	n	n	n

Instruction Fields:

`ff` - flags field:
`00` - clear `SF` and clear `DF`
`01` - clear `DF`
`10` - clear `SF`
`11` - no clear

nnnnnnnn - loop size
00000000 - empty loop: forbidden value
00000001 - 1 instruction in the loop
00000010 - 2 instructions in the loop
...
11111111 - 255 instructions in the loop

LSL1 Logical Shift Left by 1 Bit

Operation:

GReg[r]:{b30,...,b1,b0,0} ← GReg[r]:{b31,b30,...,b1,b0}

Assembler:

Syntax: `lsl1 r`

Example: `lsl1 2`
multiplies by 2 the value in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the left. The right bit (bit 0) is set to 0. No overflow is detected by the hardware.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	1

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

LSR1

Logical Shift Right by 1 Bit

Operation:

`GReg[r]:{0,b31,b30,...,b1}` ← `GReg[r]:{b31,b30,...,b1,b0}`

Syntax: `lsr1 r`

Example: `lsr1 4`
divides by 2 the unsigned value contained in GReg[4]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the right. The left bit (bit 31) is set to 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	1

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

MOV

Logical Move

Operation:

GReg[r] ← GReg[s]

Assembler:

Syntax: `mov r,s`

Example: `mov 4,0`
copies GReg[0] to GReg[4]

CPU Flags: Unaffected

Cycles: 1

Description: Move the contents of the source General Register s to the destination General Register r.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	0	1	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

NOTIFY

Notify to MCU

Operation:

```
if (jjj&4 == 0) {
            if (jjj&2 == 2) HE[CCR] ← 0
            if (jjj&1 == 1) HI[CCR] ← 1
}
else if (jjj == 4) EP[CCR] ← 0
```

(CCR stands for Current Channel Register)

Assembler:

Syntax: `notify jjj`

Example: `notify 3`
clears the HE bit for the current channel and sends an interrupt to the Host for the current channel

CPU Flags: Unaffected

Cycles: 1

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required, sends an interrupt to the corresponding CPU by setting the appropriate flag if required (HI for the corresponding channel number).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	1

Instruction Fields:

`jjj` - Channel Flags field:
`000` - unused
`001` - set HI for the current channel
`010` - clear HE for the current channel
`011` - clear HE, set HI for the current channel
`100` - clear EP for the current channel
`101` - RESERVED
`110` - RESERVED
`111` - RESERVED

OR

Logical OR

Operation:

GReg[r] ← GReg[s] | GReg[r]

Assembler:

Syntax: `or r,s`

Example: `or 3,6`
ORs GReg[3] and GReg[6] and stores the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	1	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

ORI

Logical OR with Immediate Value

Operation:

GReg[r] ← GReg[r] | immediate

Assembler:

Syntax: ori r,immediate

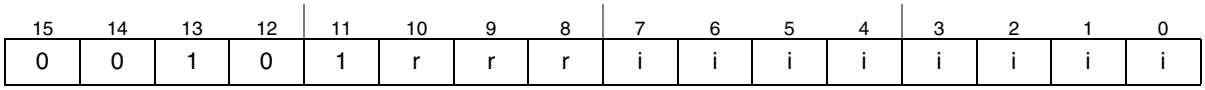
Example: ori 1,56
ORs GReg[1] and the decimal value 56 and stores the result in GReg[1]

CPU Flags: unaffected

Cycles: 1

Description: Performs an OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:
000 - GReg[0]
001 - GReg[1]
010 - GReg[2]
011 - GReg[3]
100 - GReg[4]
101 - GReg[5]
110 - GReg[6]
111 - GReg[7]

iiiiiiii - immediate value:
00000000 - 0
00000001 - 1
...
11111110 - 254
11111111 - 255

RET

Return from Subroutine

Operation:

PC ← RPC

Assembler:

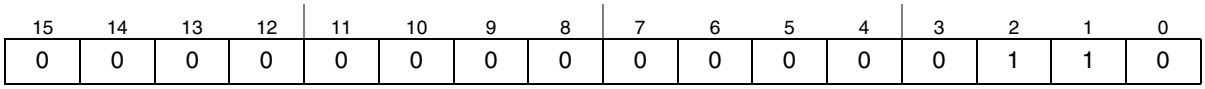
Syntax: `ret`

CPU Flags: Unaffected

Cycles: 2

Description: Return from subroutine.

Instruction Format:



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Instruction Fields: None

REVB

Reverse Byte Order

Operation:

`GReg[r]:{B3,B2,B1,B0} ← GReg[r]:{B0,B1,B2,B3}`

Assembler:

`Syntax:  revb r`

Example: `revb 5`
reverses bytes order in GReg[5]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse the byte order of any General Register.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	0

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

REVBLO

Reverse Low Order Bytes

Operation:

GReg[r]:{B3,B2,B0,B1} ← GReg[r]:{B3,B2,B1,B0}

Assembler:

Syntax: `revblo r`

Example: `revblo 0`
reverses low order bytes in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse both low order bytes of any General Register.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	1

Instruction Fields:

`rrr` - register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

ROR1

Rotate Right by 1 Bit

Operation:

`GReg[r]:{b0,b31,b30,...,b1}` ← `GReg[r]:{b31,b30,...,b1,b0}`

Assembler:

Syntax: `ror1 r`

Example: `ror1 3`
rotates bits to the right in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Rotate the bits of any General Register to the right.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	0

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

RORB

Rotate Right by 1 Byte

Operation:

`GReg[r]:{B0,B3,B2,B1} ← GReg[r]:{B3,B2,B1,B0}`

Assembler:

Syntax: `rorb r`

Example: `rorb 2`
rotates bytes to the right in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Rotate the bytes of any General Register to the right.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	1	0

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

SOFTBKPT Software Breakpoint

Operation:

Stops the current script and enters debug mode

Assembler:

Syntax: softbkpt

CPU Flags: Unaffected

Cycles:

Description: When the core executes this instruction, it has the same effect as receiving a debug request from the OnCE or via the external debug request input: the script execution halts, the PCU enters its debug state and waits for the OnCE commands that are described in Section 52.18.3, OnCE and Real-Time Debug."

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Instruction Fields: None

ST

Store Register

Operation:

```
[GReg[b] + displacement] ← GReg[r]
if (transfer_error) DF ← 1
else DF ← 0
```

Assembler:

Syntax: `st r,(b,displacement)`

Example: `st 7,(0,9)`
stores the value from GReg[7] into memory at address obtained by adding decimal value 9 to GReg[0]

CPU Flags: DF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to store on the DM bus. The data sent on the bus comes from the source General Register r. If an error occurs during the transfer, the flag DF is set, else it is cleared.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	r	r	r	d	d	d	d	d	b	b	b

Instruction Fields:

`rrr` - source register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
...
`111` - `GReg[7]`

`bbb` - base address register field:
`000` - `GReg[0]`
...
`110` - `GReg[6]`
`111` - `GReg[7]`

`ddddd` - displacement value:
`00000` - 0
`00001` - 1
...
`11111` - 31

STF

Store Register in Functional Unit

Operation:

```
[fu_address] ← GReg[r]
if (transfer_error) DF ← 1
else DF ← 0
```

fu_address is an 8-bit field

Assembler:

Syntax: `stf r,fu_address`

Example: `stf 3,0x2B`
stores the 32-bit contents of GReg[3] to the Burst DMA register MD; waits until the flush to external memory is completed

CPU Flags: DF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

Description: Sends an 8-bit address on the Functional Unit Bus (FU bus) and sends the contents of the source General Register r on the bus. If an error occurs during the transfer, the flag DF is set, else it is cleared.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the STF instruction usage with each functional unit:

- Section 52.18.1.5, Burst DMA Write (stf)" for Burst DMA
- Section 52.18.2.5, Burst DMA2 Write (stf) for Burst DMA2

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`ffffffff` - functional unit destination register and action (unspecified values are reserved):
`00000000` - MSA in incremented mode

00000100 - MDA in incremented mode
00001001 - MD byte
00001010 - MD halfword
00001011 - MD word
00001100 - clear MS error flag
00001111 - MS
00010000 - MSA in frozen mode
00010100 - MDA in frozen mode
00011000 - MD in copy mode - number of words in rrr
00100000 - MSA in incremented mode - start prefetch
00101000 - MD no data - flush
00101001 - MD byte - flush
00101010 - MD halfword - flush
00101011 - MD word - flush
00110000 - MSA in frozen mode - start prefetch
01000000 - DSA in incremented mode
01000100 - DDA in incremented mode
01001001 - DD byte
01001010 - DD halfword
01001011 - DD word
01001100 - clear DS error flag
01001111 - DS
01010001 - DSA in frozen mode - 8-bit data width
01010010 - DSA in frozen mode - 16-bit data width
01010011 - DSA in frozen mode - 32-bit data width
01010101 - DDA in frozen mode - 8-bit data width
01010110 - DDA in frozen mode - 16-bit data width
01010111 - DDA in frozen mode - 32-bit data width
01011000 - DD in copy mode - number of data in rrr
01100000 - DSA in incremented mode - prefetch data
01101000 - DD no data - flush
01101001 - DD byte - flush
01101010 - DD halfword - flush
01101011 - DD word - flush
01110001 - DSA in frozen mode - 8-bit data width - prefetch data
01110010 - DSA in frozen mode - 16-bit data width - prefetch data
01110011 - DSA in frozen mode - 32-bit data width - prefetch data
11000001 - PSA in frozen mode - 8-bit data width
11000010 - PSA in frozen mode - 16-bit data width
11000011 - PSA in frozen mode - 32-bit data width
11000101 - PSA in incremented mode - 8-bit data width
11000110 - PSA in incremented mode - 16-bit data width
11000111 - PSA in incremented mode - 32-bit data width
11001000 - PD
11001001 - PSA in decremented mode - 8-bit data width

11001010 - PSA in decremented mode - 16-bit data width
11001011 - PSA in decremented mode - 32-bit data width
11001100 - clear PS error flag
11001101 - PSA data width becomes 8-bit
11001110 - PSA data width becomes 16-bit
11001111 - PSA data width becomes 32-bit
11010001 - PDA in frozen mode - 8-bit data width
11010010 - PDA in frozen mode - 16-bit data width
11010011 - PDA in frozen mode - 32-bit data width
11010101 - PDA in incremented mode - 8-bit data width
11010110 - PDA in incremented mode - 16-bit data width
11010111 - PDA in incremented mode - 32-bit data width
11011001 - PDA in decremented mode - 8-bit data width
11011010 - PDA in decremented mode - 16-bit data width
11011011 - PDA in decremented mode - 32-bit data width
11011101 - PDA data width becomes 8-bit
11011110 - PDA data width becomes 16-bit
11011111 - PDA data width becomes 32-bit
11100001 - PSA in frozen mode - 8-bit data width - prefetch data
11100010 - PSA in frozen mode - 16-bit data width - prefetch data
11100011 - PSA in frozen mode - 32-bit data width - prefetch data
11100101 - PSA in incremented mode - 8-bit data width - prefetch data
11100110 - PSA in incremented mode - 16-bit data width - prefetch data
11100111 - PSA in incremented mode - 32-bit data width - prefetch data
11101001 - PSA in decremented mode - 8-bit data width - prefetch data
11101010 - PSA in decremented mode - 16-bit data width - prefetch data
11101011 - PSA in decremented mode - 32-bit data width - prefetch data
11101101 - PSA data width becomes 8-bit - prefetch data
11101110 - PSA data width becomes 16-bit - prefetch data
11101111 - PSA data width becomes 32-bit - prefetch data
11111111 - PS

SUB

Subtract

Operation:

```
GReg[r] ← GReg[r] - GReg[s]
T ← (GReg[r] == 0)
```

Assembler:

Syntax: `sub r,s`

Example: `sub 4,7`
SUBtracts GReg[7] from GReg[4] and stores the result in GReg[4]

CPU Flags: T

Cycles: 1

Description: Subtracts the source General Register s from the destination General Register r, and stores the result in the destination General Register r. The T flag is set if the result of the operation is 0; it is cleared if the result is not 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	0	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`sss` - source register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

SUBI

Subtract with Immediate

Operation:

`GReg[r] ← GReg[r] - immediate T ← (GReg[r] == 0)`

Assembler:

Syntax: `sub r,immediate`

Example: `sub 1,255`
SUBtracts decimal value 255 from GReg[1] and stores the result in GReg[1]

CPU Flags: T

Cycles: 1

Description: Subtracts a 0-extended 8-bit immediate value from a General Register; stores the result in the General Register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

TST

Test with Zero

Operation:

```
T ← ((GReg[s] & GReg[r]) != 0)
```

Assembler:

Syntax: `tst r,s`

Example: `tst 2,3`
ANDs GReg[2] and GReg[3] and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of the source General Register s and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	0	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`sss` - source register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

TSTI

Test Immediate

Operation:

```
T ← ((GReg[r] & immediate) != 0)
```

Assembler:

Syntax: `tsti r,immediate`

Example: `tsti 5,13`
ANDs GReg[5] and decimal value 13 and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of a 0-extended 8-bit immediate value and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

XOR

Logical Exclusive OR

Operation:

GReg[r] ← GReg[s] ^ GReg[r]

Assembler:

Syntax: `xor r,s`

Example: `xor 0,3`
XORs GReg[0] and GReg[3] and stores the result in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the eXclusive OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	0	s	s	s

Instruction Fields:

`rrr` - destination register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

`sss` - source register field:
`000` - GReg[0]
`001` - GReg[1]
`010` - GReg[2]
`011` - GReg[3]
`100` - GReg[4]
`101` - GReg[5]
`110` - GReg[6]
`111` - GReg[7]

XORI

Exclusive OR with Immediate

Operation:

`GReg[r] ← GReg[r] ^ immediate`

Assembler:

`Syntax:` `xori r,immediate`

Example: `xor 7,5`
XORs GReg[5] and decimal value 5 and stores the result in GReg[7]

CPU Flags: Unaffected

Cycles: 1

Description: Performs an eXclusive OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

`rrr` - register field:
`000` - `GReg[0]`
`001` - `GReg[1]`
`010` - `GReg[2]`
`011` - `GReg[3]`
`100` - `GReg[4]`
`101` - `GReg[5]`
`110` - `GReg[6]`
`111` - `GReg[7]`

`iiiiiiii` - immediate value:
`00000000` - 0
`00000001` - 1
...
`11111110` - 254
`11111111` - 255

YIELD, YIELDGE DONE, Yield

Limitations: By default, unsupported assembler syntax. Can be aliased to the corresponding `done` instructions (`yield` = `done` 0; `yieldge` = `done` 1). Refer to the `done` instruction description on page 52-190.

52.22 Software Restrictions

52.22.1 Unsupported Burst DMA Access Sequence

The SDMA does not support triggering a pre-fetch followed by a flush of the Burst DMA without reading or writing any data. If the flush occurs while the background pre-fetch DMA operation is still in progress, it could result in un-defined behavior. An example of the sequence which could result in undefined results is in **Example 52-10**.

Example 52-10. Instruction sequence not supported

```
stf r1, MSA|PF    ; Update source address, triggers data pre-fetch in the background
mov R0,R0         ; Execute multiple assembly instructions, none of which read
mov R0,R0         ; or write data to/from MD
stf MD|SZ0|FL     ; Flush FIFO without writing data. If the pre-fetch is still in
                  ; progress when this instruction is executed, there could be
                  ; undefined operation.
```

A work-around to avoid any undesirable results is to first read MD to ensure the pre-fetch is complete before the flush is attempted.

Example 52-11. Work-Around to Example 52-10

```
stf r1, MSA|PF    ; Update source address, triggers data pre-fetch
mov R0,R0         ; Execute multiple assembly instructions, none of which read
mov R0,R0         ; or write data to/from MD
ldf r2, MD        ; dummy read of MD to ensure pre-fetch is complete before the
                  ; next instruction
stf MD|SZ0|FL     ; Flush FIFO without writing data.
```

52.23 Application Notes

52.23.1 Data Structures for Boot Code and Channel Scripts

SDMA boot code downloads the different channel contexts and the scripts that will be executed on SDMA channels during the application. The boot code is run after reset when channel 0 is started by the AP. The boot code is also known as channel 0 script.

The boot code is based on the Channel Control Block (CCB) and Buffer Descriptor (BD) mechanisms that are data structures located into the AP memory space. With these data structures, it is possible to instruct SDMA to download scripts and contexts but also to dump a context or a script to a destination data buffer. Channel scripts also use the CCB and BD data structures to pass instructions and/or pointers to data to be copied.

The format, processing, and field definition of the CCB and BD are defined and performed entirely by the software script rather than the SDMA hardware. An overview of the format and structure is provided here, but for complete details refer to the SDMA software documentation of Section 52.25, References."

The CCB and BD data structures are accessed by SDMA using DMA and processed by the SDMA scripts. The ROM contains common sub-routines for processing these data structures which may be called by the bootload and channel scripts.

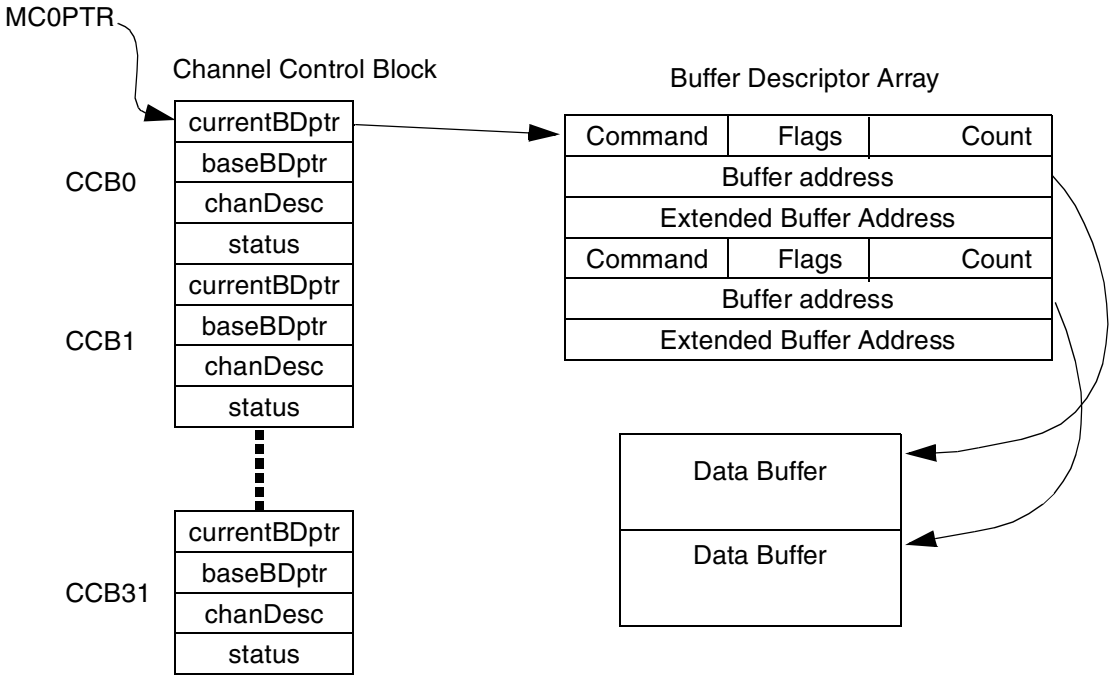


Figure 52-80. Data Structures Layout

Figure 52-80 shows an example how these data structures are linked to pass command and pointers to data buffers. The SDMA's MC0PTR register holds the base address of the Channel 0 Control Block (CCB0). The Channel 0 control block holds a pointer to the array of buffer descriptors. The buffer descriptors are used to tell the channel 0 (boot channel) what to do as described Section 52.23.1.1, Buffer Descriptor Format."

52.23.1.1 Buffer Descriptor Format

Buffer descriptors are three longs (32-bit words) in size as shown in Figure 52-81. A buffer descriptor describes the properties of the data buffer it points to. The buffer descriptors can be used for linear or circular data buffers in the AP processor memory. The CCB contains a pointer to the base BD as well as the current BD.

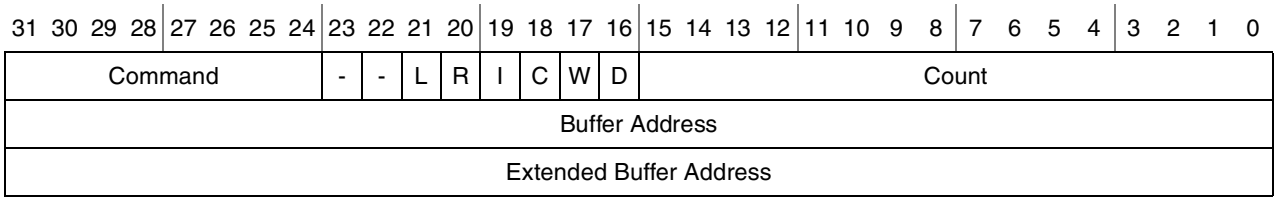


Figure 52-81. Buffer Descriptor

Table 52-96 summarizes the definitions of fields in the buffer descriptor.

Table 52-96. Buffer Descriptor Field Descriptions

Field	Description
31–24 Command	Command. The command field is used to differentiate operations performed within a script when the script accesses this particular buffer descriptor. The use of this field can be defined by the script. The command values defined for the bootload script are defined in Section 52.23.1.2, Buffer Descriptor Commands for Bootload scripts." Refer to the individual script definition in script library documents in Section 52.25, References" for command field definitions for other scripts.
23–22	Reserved
21 L	Last Buffer Descriptor: This bit is set in SDMA IPC scripts to indicate to the receiving Core that the transfer has ended. Whenever the source finishes transferring the count it wanted to transfer, it sets LAST_BIT in the destination BD, to let the destination know that transfer is over .This bit also tells the destination software that when it processes the destination BDs, they need not process any BD after the BD with the LAST_BIT set.For example, when the DSP prepares a single buffer descriptor with count equals to 25 and MCU prepares a single buffer descriptor with count equals 100. When 25 bytes have been transferred from DSP to MCU, the DSP buffer descriptor is normally closed while the MCU buffer descriptor will have the L bit set and the byte count updated to 25.
20 R	erroR. Indicates an error occurred on the channel's buffer descriptor requested command. Some scripts may overwrite the command field with an error code indicating the source of the error. 0 No Error 1 Error
19 I	Interrupt. When SDMA has finished to process data transfer attached to this buffer descriptor, send an interrupt to the AP. 0 No Interrupt 1 Interrupt the processor when BD is complete
18 C	CONTinuous. This buffer is allowed to receive multiple transmit buffers or is allowed to transmit to multiple receive buffers.The Continuos bit is decoded at the end of the processing of a BD to determine if the SDMA script must open a new BD to potentially continue the data transfer. 0 No further buffer descriptors 1 SDMA should move to the next Buffer descriptor after this one
17 W	Wrap. Indicates if this buffer descriptor is the last one for the channel control block. When encountering this bit set, the SDMA scripts updates the CurrentBD pointer to point to the first Buffer Descriptor of the array. This bit is set if the AP wants to organize the array of BD in a circular way (like a ring). When all BD have been processed and if Wrap bit and CONtinuous bit are set in the last BD, the SDMA script will wrap around and it will try to re-open the first BD. 0 No Error 1 Wrap to first buffer descriptor after this one is processed.
16 D	D - "Done": bit 16: indicates the "ownership" of the buffer descriptor. When D=0 the host owns the buffer descriptor; when D=1 SDMA owns the buffer descriptor. In the case of the channel 0, D=1 indicates the SDMA has not yet processed this buffer, D=0 indicates the SDMA has processed this buffer. 0 AP owns the buffer. 1 SDMA owns the buffer

Table 52-96. Buffer Descriptor Field Descriptions

Field	Description
15–0 Count	Count. the count field (bit 15-0) indicates the size of the data to be transmitted, the size of the data buffer pointed to by the buffer descriptor. The SDMA memory structure is different for program memory (16-bits shorts/half-words) and data memory (32-bits long). For channel 0 buffer descriptors, Count is expressed in 16-bit half-words when PM is addressed and in 32-bit words when DM is addressed. Count is typically expressed in bytes for other channel scripts, but the unit is dependant on the script.
31–0	Buffer address. Address pointer to the data buffer.
31–0	Extended buffer address. Additional pointer or other information required by some scripts.

The buffer descriptors form an array of programmable size. If the last buffer descriptor is marked by the Wrap flag-bit W=1, the array of buffer descriptor is treated as a ring with some logically continuous portion owned by the AP with D=0, and the remainder owned by the SDMA with D=1. The count field of the buffer descriptor indicates how much data has been transmitted.

If AP has prepared 3 buffers to be filled by the SDMA script, it has also prepared 3 BD, one for each buffer. The *Cont* and *Wrap* bits are used to organize the buffers in a circular way. For example, *CONT*inous bit is set to 1 in the 2 first BDs and *Wrap* is set in the 3rd BD. The SDMA script opens and processes BD#1. Since *CONT*inous bit is set for this BD, the SDMA will open the second BD and it will process it. Each time a BD is processed, its *Done* bit is reset by the SDMA. After the 3rd BD, if *CONT*inous is not set but if *Wrap* is set, the SDMA script stops here and the next time the channel will be triggered, the script will open the BD pointed by the currentBDptr pointer of the CCB and it will correspond to the first buffer descriptor.

If the *CONT*inous bit and *Wrap* bits are both set in the 3rd BD, the script will close it and it will try to open the first BD. An error may occur at this point if the BD#1 has already been processed and its *Done* bit is 0. The SDMA script cannot process a BD with a *Done* bit to 0. It means the BD is not ready to be processed. To avoid this situation, the *CONT*inous bit should not be set for the last BD if *Wrap* is set, and the Interrupt flag must set for the last BD. It will warn the owner of the BD that all the BDs have been processed and it has to re-set to 1 the *Done* bit of all the BD's if it desires the SDMA to fill them again. Basically, if the AP expects the SDMA to fill up the buffers in a circular fashion, then it's the responsibility of the AP to set the *Done* bit of a buffer descriptor at an appropriate time.

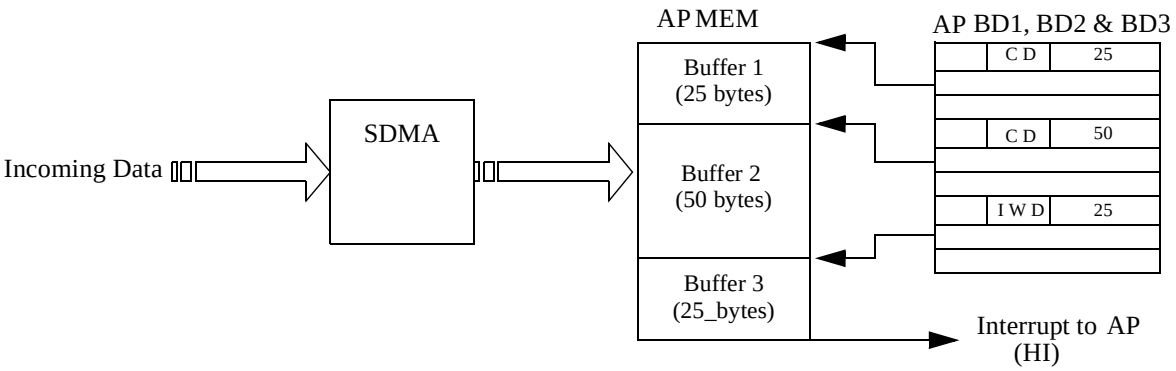


Figure 52-82. Buffer Descriptor Flow

Figure 52-82 shows an example buffer descriptor flow. When the incoming data is stored and fills the first buffer of 25 bytes, the SDMA script opens the second BD because the *CONT*inuous bit was set. Then next incoming data is put in the second buffer. After receiving 50 bytes, the second buffer descriptor is also closed. The *Done* bit is reset and the third BD is opened. After receiving another 25 bytes, the third buffer is full and an interrupt is sent to the AP because the Interrupt flag is set in the 3rd BD. The *CONT*inuous flag is not present the transfer is over. The next time the script will be triggered, the BD to be opened will be the first buffer descriptor since the *Wrap* flag was set in the 3rd BD. It is the AP responsibility to set the *Done* bit of all the BD if it wants to use the same buffers.

52.23.1.2 Buffer Descriptor Commands for Bootload scripts

The command field of the buffer descriptor is defined separately for each script. Table 52-97 lists the buffer descriptor commands defined for the channel 0 bootload script.

Table 52-97. Channel Zero Buffer Descriptor Commands

Command Field (binary)	Command	Description	Buffer Address	Extended Buffer Address
0000_0001 (0x01)	C0_SET_DM	Load SDMA data memory (RAM) from AP memory buffer	AP memory source address	SDMA memory destination address
0000_0010 (0x02)	C0_GET_DM	Copy SDMA data memory (RAM) to AP memory buffer	AP memory destination address	SDMA memory source address
0000_0100 (0x04)	C0_SET_PM	Load SDMA program memory (RAM) from AP memory buffer	AP memory source address	SDMA memory destination address
0000_0110 (0x06)	C0_GET_PM	Copy SDMA program memory (RAM) to AP memory buffer	AP memory destination address	SDMA memory source address
cccc_c111 (0x07 | CHN)	C0_SETCTX	Load Context for channel ccccc into SDMA RAM from AP memory buffer	AP memory source address	—
cccc_c011 (0x03 | CHN)	C0_GETCTXT	Copy Context for channel ccccc from SDMA RAM to AP memory buffer	AP memory destination address	—

The Channel 0 bootload commands are summarized as follows:

- C0_SET_[PM-DM]: load the buffer descriptor data in the SDMA local memory at the address pointed to by the “extended buffer address” field. The SDMA RAM can be seen as a Program Memory (PM, 16-bit address) or Data Memory (DM 32-bit address). When C0_SET_PM is used, the count field is expressed in “shorts” (16-bit half words), this command can be used to download scripts. When C0_SET_DM is used, the count field is expressed in “long” (32-bit words), this command can be used to download channel contexts to the context channel area in RAM.
- C0_GET_[PM-DM]: write to the buffer descriptor’s data buffer the content of the SDMA local memory from the address pointed to by the “extended buffer address” field for the length defined by the count in the buffer descriptor. C0_GET_PM is used to dump some part of the Program Memory (may be used to dump context of a channel), therefore count is expressed in “shorts”; while C0_GET_DM is used to dump to the buffer descriptor’s data buffer, so the count field is in “longs.”

- C0_SETCTX: load a context into the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using the channel number the script computes the offset of the context data pointer for the channel relative to the context page base to use as the destination address in SDMA memory. Then the C0_SET_DM command explained above is invoked to load SDMA RAM from memory. The counter indicates the size in words of the context structure.

  Command value: (in binary) cccc c111, where ccccc is the channel number (5 bits). For instance, 0x0F means set context for channel 1, 0xFF means set context for channel 31.
- C0_GETCTX: write to the buffer descriptor's data buffer the content of the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using this channel number, the script computes the offset of the context data pointer for the channel relative to the context page base to use as the source address for the copy. Then the C0_GET_DM command explained above is invoked to copy the context to memory. The counter indicates the size in words of the context structure.

  Command value: (in binary): cccc c011, where ccccc is the channel number (5 bits). For instance, 0x03 means get context of channel 1, 0xFB means get context of channel 31.

NOTE

To download channel context, C0_SETDM and C0_SETCTXT command can be used but the second one is easier because the channel number is embedded into the command field, whereas with the C0_SETDM, the pointer to the channel context area must be written into the extended buffer address field of the buffer descriptor.

52.23.1.3 Example of Buffer Descriptors for Channel 0.

Figure 52-84 illustrates the buffer descriptors that must be set in AP memory space, before execution of boot code, to download contexts and scripts of channels 1, 4, and 10. After boot code execution, SDMA memory will be populated with the different contexts and scripts as presented in Figure 52-83.

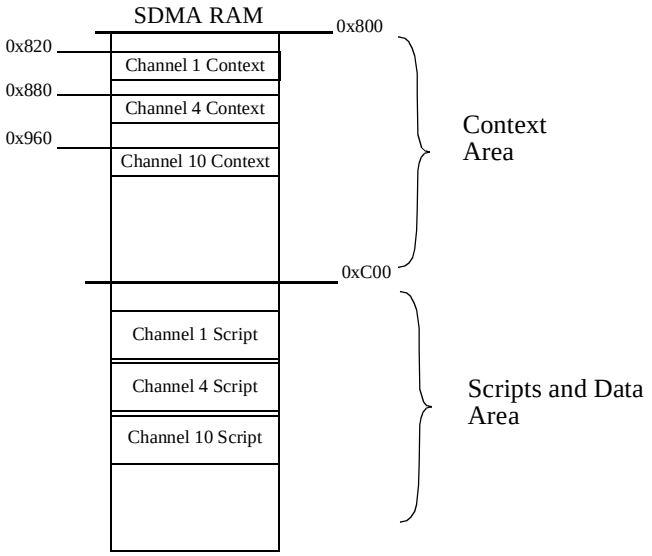


Figure 52-83. Example of SDMA RAM After Boot Session

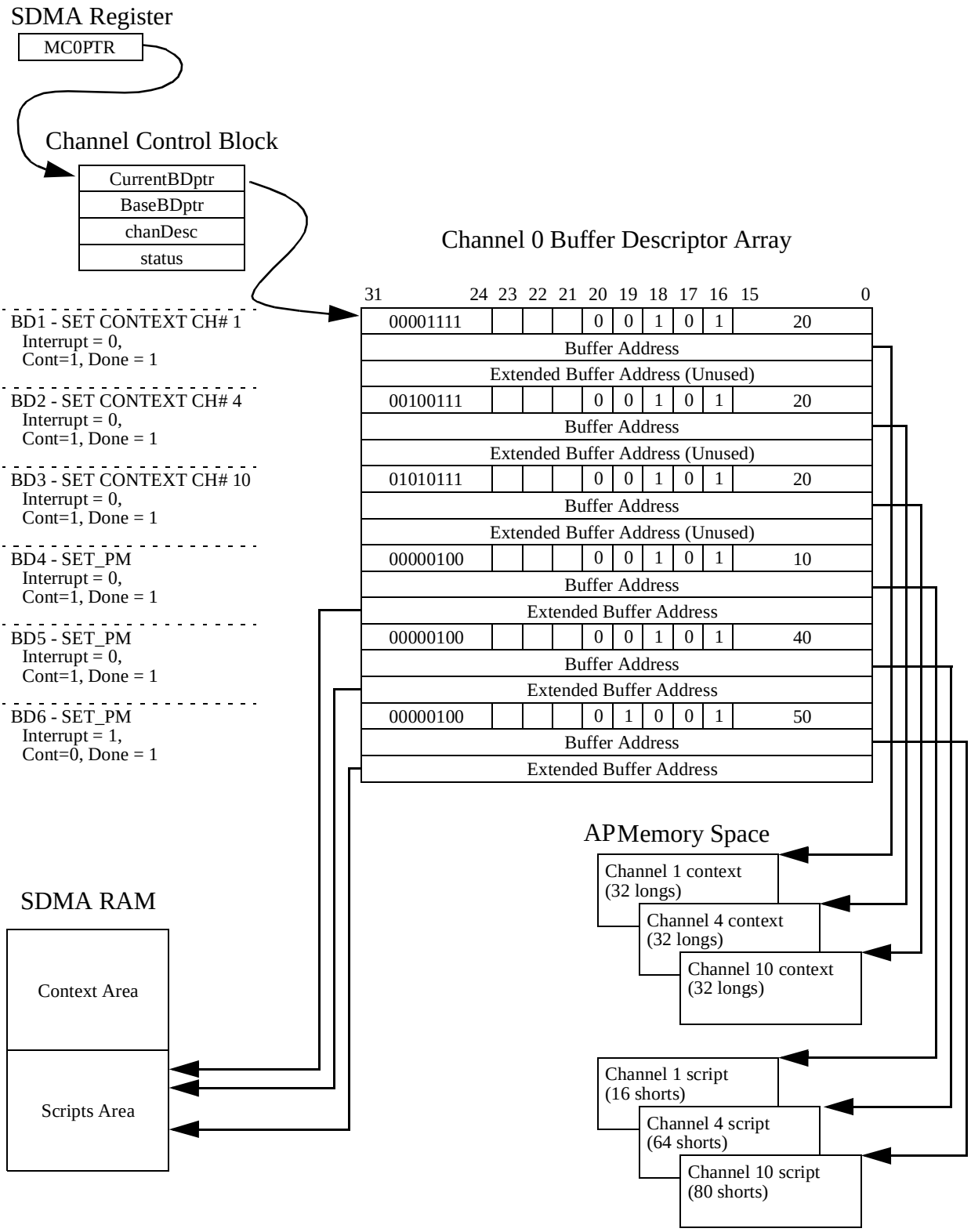


Figure 52-84. Example of Buffer Descriptors for 3 Scripts and 3 Contexts

52.23.1.4 Channel Context

There are 32 channel context memory structures pointed to by the local save area pointer. These channel context memory structures are fixed. The script in the SDMA computes the memory offset for a given channel based on the structure length and channel number. Figure 52-85 shows the structure of the channel context as it is saved in the SDMA local memory (RAM).

A channel context consists in 24 words, one per register. A total of 32 words are reserved for every channel. The additional 8 words are called scratch ram and they are dedicated to each channel. This memory area is commonly used for stack management.

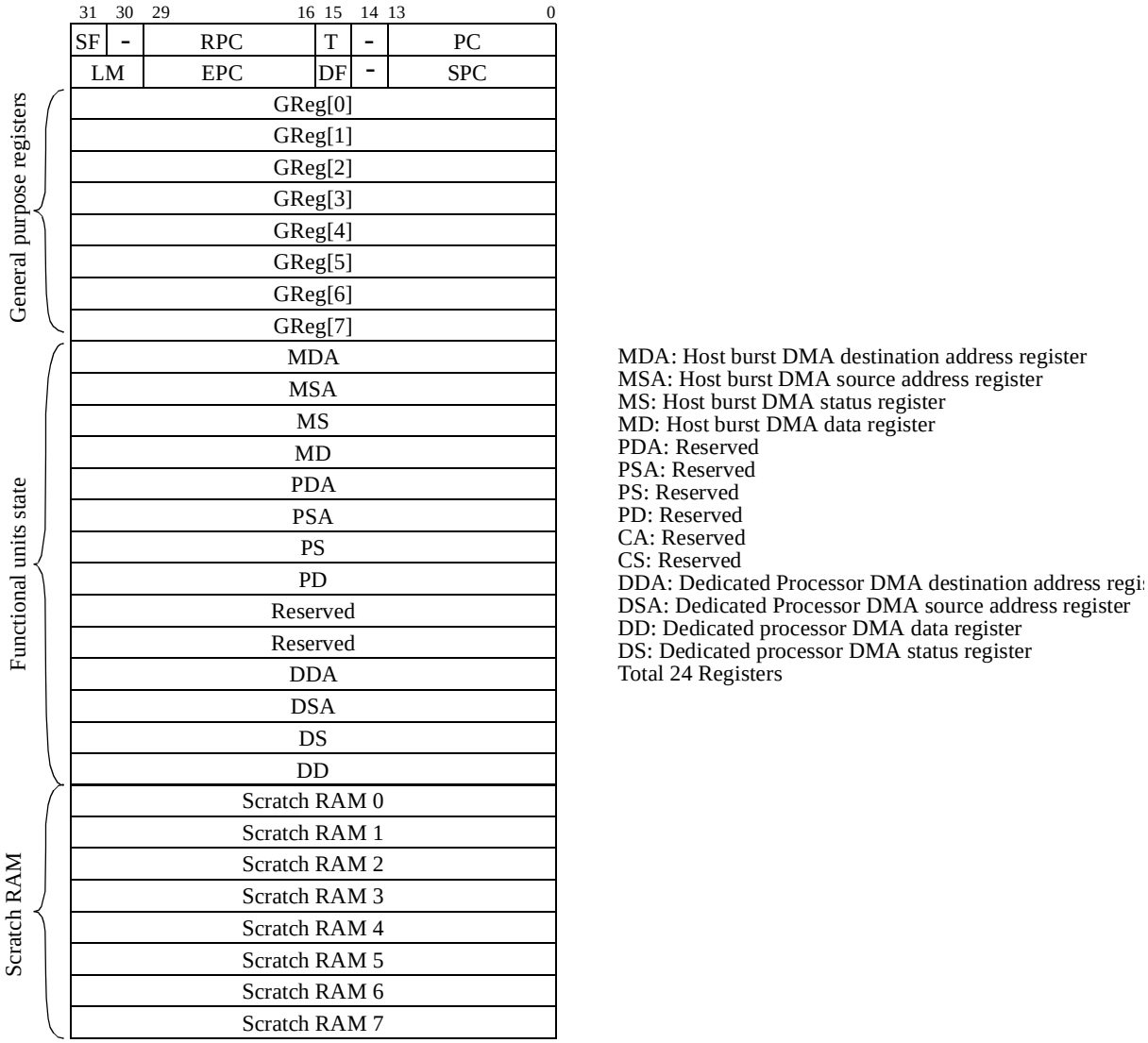


Figure 52-85. Channel Context Memory Structure

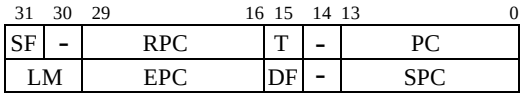
The structure is divided in 4 areas:

- Channel status registers
- General purpose registers

- Functional units state registers reflecting the state of the AP DMAs (Burst and Burst DMA2).
- Scratch RAM

The details of the channel context status registers are described in Figure 52-86.

The PC field of the first long register must point to the SDMA RAM address where the script that will be executed on the channel is located and this value equals the one stored in the extended buffer address of the buffer descriptor with C0_SETPM command.



SF: Source fault while loading data
RPC: Return program counter
T: Test bit: status of arithmetic and test instructions
PC: Program counter
LM: Loop mode
EPC: Loop end program counter
DF: Destination fault while storing data
SPC: Loop Start program counter

Figure 52-86. SDMA State Registers (ShPC, ShLoop)

52.24 Definitions, Acronyms, Abbreviations

channel	Refers to a defined path for flow of data between a source and sink. A channel is uni-directional. Example source and sink nodes are AP Processor or Peripherals. Up to 32 channels can be defined and operating at any one time.
context switch	Saving micro RISC (for example, SDMA core) registers by copying their contents into internal SRAM space followed by a load of the registers from a different channel. Essentially, a context switch means that the current state of the SDMA micro-RISC engine is saved, and the new state of the core is determined by loading the core registers with the state of a different channel.
script	An assembly language program executed by the SDMA core. Scripts reside in the ROM or may be pre-loaded into the SDMA SRAM.
Virtual DMA	DMA function executed as observed from the outside. However, the function is actually executed by a script in the micro-RISC core.

52.25 References

The following references provide useful information.

- i.MX51 SDMA Scripts User Manual
- SDMA Scripts Library—Software Design Specification
  Rev 5.0, Feb 21, 2005
  Doc number: 04-1779-SDS-ZFR11
  External Link: https://www.freescale.com/cgi/doc/154727497/SDMA_SCRIPTS.doc

Chapter 53
Sony/Philips Digital Interface Transmitter (SPDIF Tx)

53.1 Introduction

The Sony/Philips Digital Interface Transmitter (SPDIF Tx) audio module is a stereo transmitter that allows the processor to transmit digital audio over it. Figure 53-1 shows a block diagram of the SPDIF transmitter data paths and its interface.

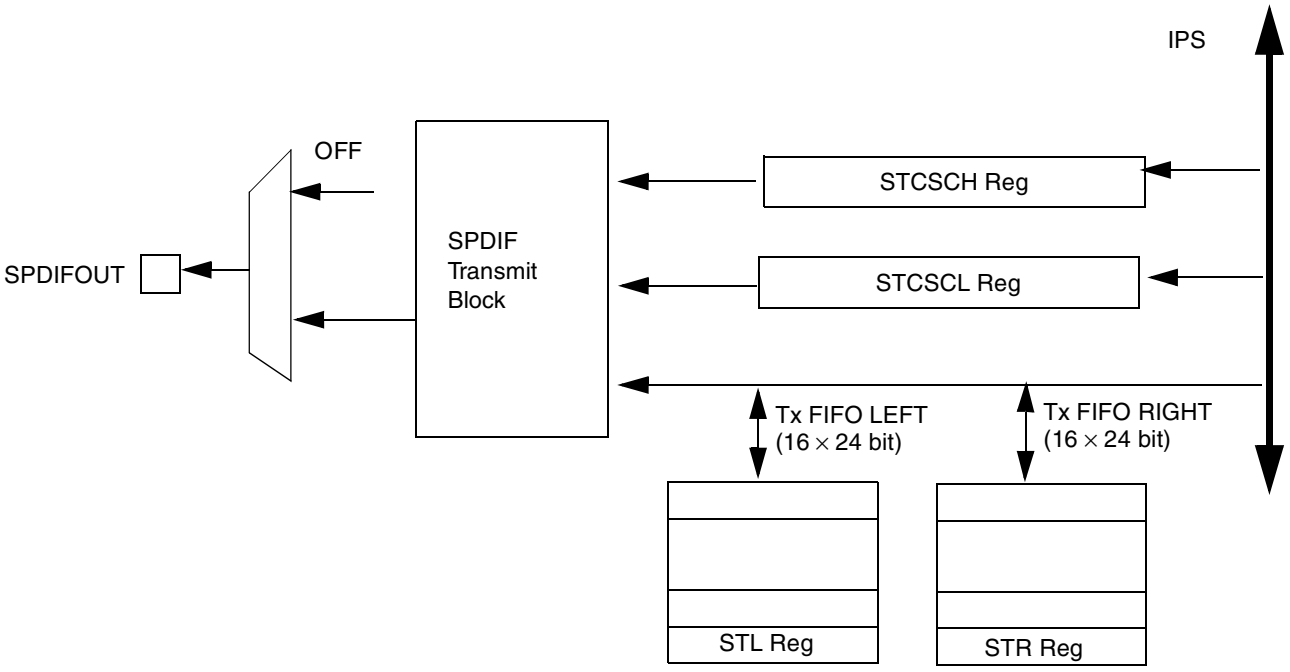


Figure 53-1. SPDIF Transmitter Data Interface Block Diagram

53.1.1 Overview

For the SPDIF transmitter, the audio data is provided by the processor via the STL and STR registers. The channel status is also provided via the corresponding registers. Zero is always inserted in the user data. The SPDIF transmitter generates a SPDIF output bitstream in the biphase mark format (IEC958), which consists of audio data, channel status, and user data.

In the SPDIF transmitter, the IEC958 biphase bit stream is generated on both edges of the SPDIF transmit clock. The SPDIF transmit clock is generated by the SPDIF internal clock generate module and the sources are from outside of the SPDIF block.

Figure 53-2 shows the clock structure of the SPDIF transmitter.

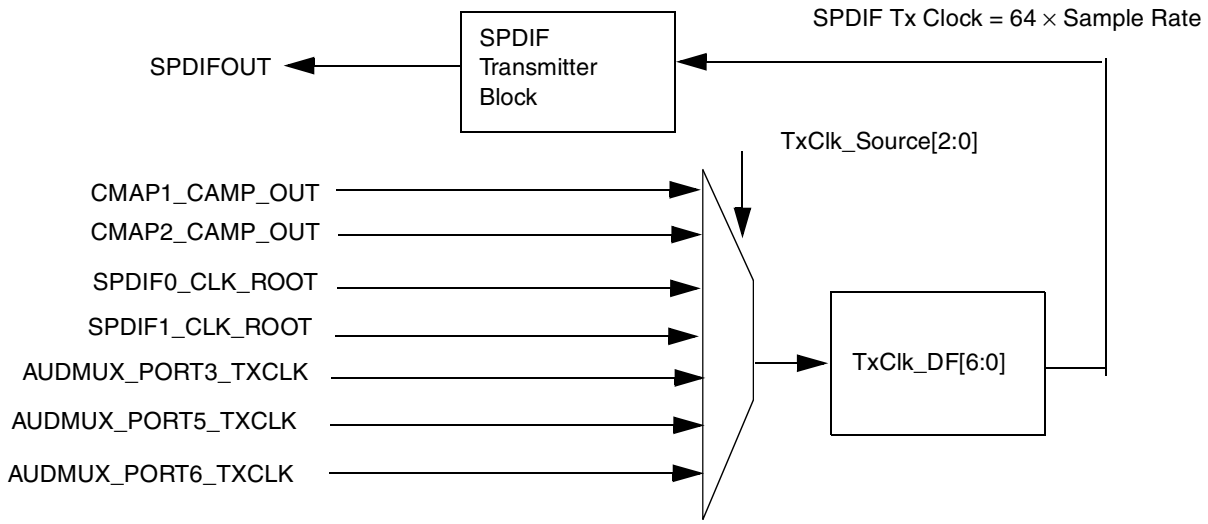


Figure 53-2. SPDIF Transmitter Clock Diagram

53.1.2 Features

The features are as follows:

- SPDIF Transmitter
  - IEC 60958 format SPDIF output
  - 7 transmit clock source
  - Consumer channel status support
- Support for interrupt and DMA

53.2 External Signal Description

Table 53-1 shows the signal properties.

Table 53-1. Signal Properties

Signal Name	Signal Type	Description
SPDIFOUT	Output	SPDIF Output Line IEC958 data in biphase mark format. (Consumer channel status).

53.3 Memory Map and Register Definition

This section contains a register map, key to register fields, register figure conventions, and a register summary.

53.3.1 Memory Map

Table 53-3 shows the SPDIF register map.

Table 53-2. SPDIF Register Map

Address	Register	Access	Reset Value
0xBASE_00(SCR)	SPDIF Configuration Register	R/W	32‘h0000_0400
0xBASE_0C(SIE)	SPDIF Interrupt Enable register	R/W	32’h0000_0000
0xBASE_10(SIS/SIC)	SPDIF Interrupt status/clear register	R-Stat W-Clear	32’h0000_0002
0xBASE_2C(STL)	SPDIF Tx Left channel data register	W	32’h0000_0000
0xBASE_30(STR)	SPDIF Tx Right channel data register	W	32’h0000_0000
0xBASE_34(STCSCH)	SPDIF Tx Consumer channel status High register	W	32’h0000_0000
0xBASE_38(STCSCL)	SPDIF Tx Consumer channel status low register	W	32’h0000_0000
0xBASE_50(STC)	SPDIF Tx Clk config register	R/W	32’h0002_0f00

53.3.2 Register Summary

Figure 53-3 shows the key to the register fields and Table 53-3 shows the register figure conventions.

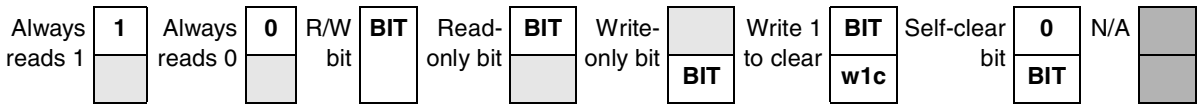


Figure 53-3. Key to Register Fields

Table 53-3. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit’s value (other than a hardware reset).

Table 53-3. Register Figure Conventions (continued)

Convention	Description
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 53-4 shows the SPDIF register summary.

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE_00 (SCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxAutoSync	TxFIFOEmpty_Sel[1]
	W																
	R	TxFIFOEmpty_Sel[0]	0	Low_Power	SW_Reset	TxFifo_Ctrl[1:0]		0	PDIR_Tx	0	0	ValCtrl	TxSel[2:0]			0	0
	W																
0xBASE_0C (SIE)	R	0	0	0	0	0	0	0	0	0	0	0	0	TxUnOv_En	TxResyn_En	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxEm_En	0
	W																
0xBASE_10 (SIS/SIC)	R	0	0	0	0	0	0	0	0	0	0	0	0	TxUnOv	TxResyn	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxEm	0
	W																

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE_2C (STL)	R	0	0	0	0	0	0	0	0	8'b0							
	W									TxDataLeft[23:16]							
	R	16'b0															
	W	TxDataLeft[15:0]															
0xBASE_30 (STR)	R	0	0	0	0	0	0	0	0	8'b0							
	W									TxDataRight[23:16]							
	R	16'b0															
	W	TxDataRight[15:0]															
0xBASE_34 (STCSCH)	R	0	0	0	0	0	0	0	0	8'b0							
	W									TxCChannelCons_h[23:16]							
	R	16'b0															
	W	TxCChannelCons_h[15:0]															
0xBASE_38 (STCSCL)	R	0	0	0	0	0	0	0	0	8'b0							
	W									TxCChannelCons_l[23:16]							
	R	16'b0															
	W	TxCChannelCons_l[15:0]															
0xBASE_50 (STC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	TxClk_Source [2:0]			0	TxClk_DF[6:0]						
	W																

Table 53-4. SPDIF Register Summary

53.3.3 Register Descriptions

This section provides the detailed register descriptions in address order.

53.3.3.1 SPDIF Configuration Register (SCR)

See Figure 53-4 for illustration of valid bits in the SPDIF Configuration Register and Table 53-5 for description of the bit fields in the register.

0x43FA_0000 (SCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxAutoSync	TxFIFOEmpty_Sel[1]
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TxFIFOEmpty_Sel[0]	0	Low_Power	SW_Reset	TxFifo_Ctrl[1:0]		0	PDIR_Tx	0	0	ValCtrl	TxSel[2:0]			0	0
W																
RESET	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Figure 53-4. SPDIF Config Register

Table 53-5. SPDIF Config Register (SCR) Field Descriptions

Field	Description
3–8	Reserved
17 TxAutoSync	0 Tx FIFO auto sync off 1 Tx FIFO auto sync on
16–15 TxFIFOEmpty_Sel[1:0]	00 Empty interrupt if 0 sample in Tx left and right FIFOs 01 Empty interrupt if at most 4 sample in Tx left and right FIFOs 10 Empty interrupt if at most 8 sample in Tx left and right FIFOs 11 Empty interrupt if at most 12 sample in Tx left and right FIFOs
14	Reserved
13 Low_Power	When 1 is written to this bit, SPDIF enters low-power mode. Read return 1 when SPDIF is in low-power mode.
12 SW_reset	When 1 is written to this bit, SPDIF software resets. Need to read the register 5 times to make it come out of the reset status. When in the reset process, it returns 1 when read. Otherwise, it returns 0 when read.

Table 53-5. SPDIF Config Register (SCR) Field Descriptions

Field	Description
11–10 TxFifo_Ctrl	00 Send out digital zero on SPDIF Tx 01 Normal operation 10 Reset to 1 sample remaining 11 Reserved
9	Reserved
8 PDIR_TX	DMA Transmit Request enable. DMA transfer starts when the PDIR_Tx bit is set, and the TxFIFO is empty compared to the Tx watermark.
7–6	Reserved
5 ValCtrl	0 Outingoing Validity always set 1 Outingoing Validity always clear
4–2 TxSel	000 Off and output 0 101 Normal operation
1–0	Reserved

53.3.3.2 SPDIF Interrupt Enable Register (SIE)

See Figure 53-5 for illustration of valid bits in the SPDIF Interrupt Enable Register and Table 53-6 for description of the bit fields in the register.

0x43FA_000C (SIE) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	TxUnOv_En	TxResyn_En	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxEm_En	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 53-5. SPDIF Interrupt Enable Register (SIE)

Table 53-6. SPDIF Interrupt Enable Register (SIE) Field Descriptions

Field	Description
31–20	Reserved
19 TxUnOv_En	SPDIF transmit FIFO under/overrun Interrupt enable bit
18 TxResyn_En	SPDIF transmit FIFO resync Interrupt enable bit
17–2	Reserved
1 TxEm_En	SPDIF transmit FIFO empty interrupt enable bit
0	Reserved

53.3.3.3 SPDIF Interrupt Status/Clear Register (SIS/SIC)

See Figure 53-6 for illustration of valid bits in the SPDIF Interrupt Status/Clear Register and Table 53-7 for description of the bit fields in the register.

0x43FA_0010 (SIS/SIC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	TxUnOv	TxResyn	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TxEm	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Figure 53-6. SPDIF Interrupt Status/Clear Register (SIS/SIC)

Table 53-7. SPDIF Interrupt Status/Clear Register (SIS/SIC) Field Descriptions

Field	Description
31–20	Reserved
19 TxUnOv	SPDIF transmit FIFO under/overrun Interrupt bit. When SPDIF transmit FIFO under/overrun interrupt happens, read 1 from the bit, else read 0. Write 1 to the bit to clear SPDIF transmit FIFO under/overrun interrupt.

Field	Description
18 TxResyn	SPDIF transmit FIFO resync Interrupt bit. When SPDIF transmit FIFO resync interrupt happens, read 1 from the bit, else read 0. write 1 to the bit to clear the SPDIF transmit FIFO resync interrupt.
17–2	Reserved.
1 TxEm	SPDIF transmit FIFO empty interrupt bit. When SPDIF transmit FIFO empty interrupt happens, read 1 from the bit, else read 0. Can't be cleared with writing 1 to the bit. To clear it, write data to Tx FIFO.
0	Reserved.

53.3.3.4 SPDIF Tx Left Channel Data Register (STL)

See Figure 53-7 for illustration of valid bits in the SPDIF Tx Left Channel Data Register and Table 53-8 for description of the bit fields in the register.

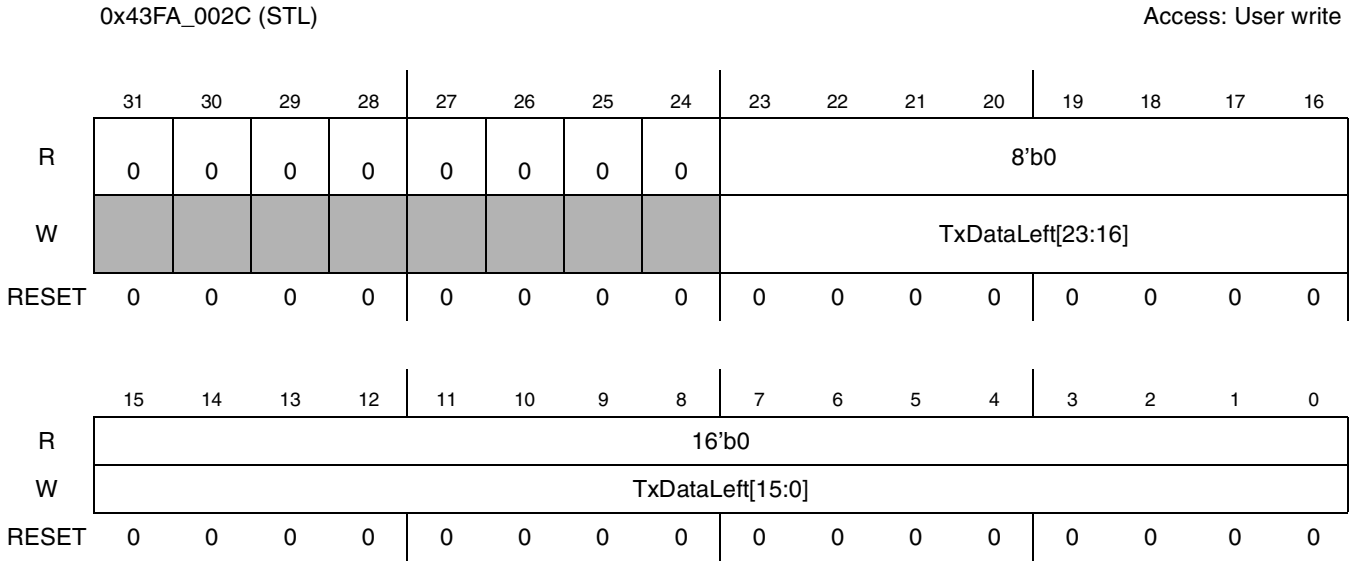


Figure 53-7. SPDIF Tx Left Channel Data Register

Table 53-8. SPDIF Tx Left Channel Data Register (STL) Field Descriptions

Field	Description
32–24	Reserved.
23–0 TxDataLeft[24:0]	SPDIF transmit left channel data. It is write-only, and always returns zeros when read

53.3.3.5 SPDIF Tx Right Channel Data Register (STR)

See Figure 53-8 for illustration of valid bits in the SPDIF Tx Right Channel Data Register and Table 53-9 for description of the bit fields in the register.

0x43FA_0030 (STR) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	8'b0							
W									TxDataRight[23:16]							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	16'b0															
W	TxDataRight[15:0]															
RESET	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Figure 53-8. SPDIF Tx Right Channel Data Register

Table 53-9. SPDIF Tx Right Channel Data Register (STR) Field Descriptions

Field	Description
32–24	Reserved.
23–0 TxDataRight[24:0]	SPDIF transmit right channel data. It is write-only, and always returns zeros when read

53.3.3.6 SPDIF Tx Consumer Channel Status High Register (STCSCH)

See Figure 53-9 for illustration of valid bits in the SPDIF Tx Consumer Channel Status High Register and Table 53-10 for description of the bit fields in the register.

0x43FA_0034 (STCSCH) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	8'b0							
W									TxCChannelCons_h[23:16]							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	16'b0															
W	TxCChannelCons_h[15:0]															
RESET	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Figure 53-9. SPDIF Tx Consumer Channel Status High Register

Table 53-10. SPDIF Tx Consumer Channel Status High Register (STCSCH) Field Descriptions

Field	Description
32–24	Reserved
23–0 TxCChannelCons_h[23:0]	SPDIF transmit Cons. C channel data, contains first 24 bits without interpretation. When read, it returns the latest data written by the processor

53.3.3.7 SPDIF Tx Consumer Channel Status Low Register (STCSCL)

See Figure 53-10 for illustration of valid bits in the SPDIF Tx Consumer Channel Status Low Register and Table 53-11 for description of the bit fields in the register.

0x43FA_0038(STCSCL) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	8'b0							
W									TxCChannelCons_l[23:16]							
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	16'b0															
W	TxCChannelCons_l[15:0]															
RESET	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

Figure 53-10. SPDIF Tx Consumer Channel Status Low Register

Table 53-11. SPDIF Tx Consumer Channel Status Low Register (STCSCL) Field Descriptions

Field	Description
32–24	Reserved.
23–0 TxCChannelCons_l[23:0]	SPDIF transmit Cons. C channel data, contains next 24 bits without interpretation. When read, it returns the latest data written by the processor

53.3.3.8 SPDIF Tx Clk Config Register (STC)

See Figure 53-11 for illustration of valid bits in the SPDIF Tx Clk Config Register and Table 53-12 for description of the bit fields in the register.

0x43FA_0050(STC) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TxClk_Source			0	TxClk_DF						
W																
RESET	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Figure 53-11. SPDIF Tx Clk Config Register

Table 53-12. SPDIF Tx Clk config Register (STC) Field Descriptions

Field	Description
31–11	Reserved
10–8 TxClk_Source[2:0]	000 CAMP1_CMAP_OUT input 001 SPDIF0_CLK_ROOT input 010 SPDIF1_CLK_ROOT input 011 AUDMUX_Port3_TXCLK input 100 AUDMUX_Port5_TXCLK input 110 AUDMUX_Port6_TXCLK input 111 CAMP2_CMAP_OUT input
7	Reserved
6–0 TxClk_DF [6:0]	Divider factor(1–128) 4'd0: divider factor is 1 4'd1: divider factor is 2 .... 4'd127: divider factor is 128

53.4 Functional Description

Audio data for the SPDIF transmitter is provided via the STL and STR registers. Clocking for SPDIF transmitter is from the AUDMUX_Port3_TXCLK signal, AUDMUX_Port5_TXCLK signal, AUDMUX_Port6_TXCLK signal, the on-chip quartz oscillator and the external clock input. A

multiplexer is used to choose the clock source. The SPDIF transmitter clock source can be divided down as needed using TxClk_DF bits in STC register. The SPDIF transmitter output can be chosen from either the SPDIF transmitter block, or disabled.

Zero is always inserted in the user data.

53.4.0.1

Chapter 54
System Reset Controller (SRC)

54.1 Introduction

The SRC generates the different reset signals for all of the modules in the i.MX51. Figure 54-1 is a block diagram of the the SRC.

54.1.1 Overview

The reset control is responsible for the generation of all resets signals and boot decoding. The reset control determines the source and the type of reset, such as POR, WARM, COLD, and performs the necessary reset qualification and stretching sequences. Based on the type of reset, the reset logic generates the reset sequence for the entire IC. Whenever the chip is powered on, the reset is isseud threw ipp_reset_b signal and the entire chip is reset.

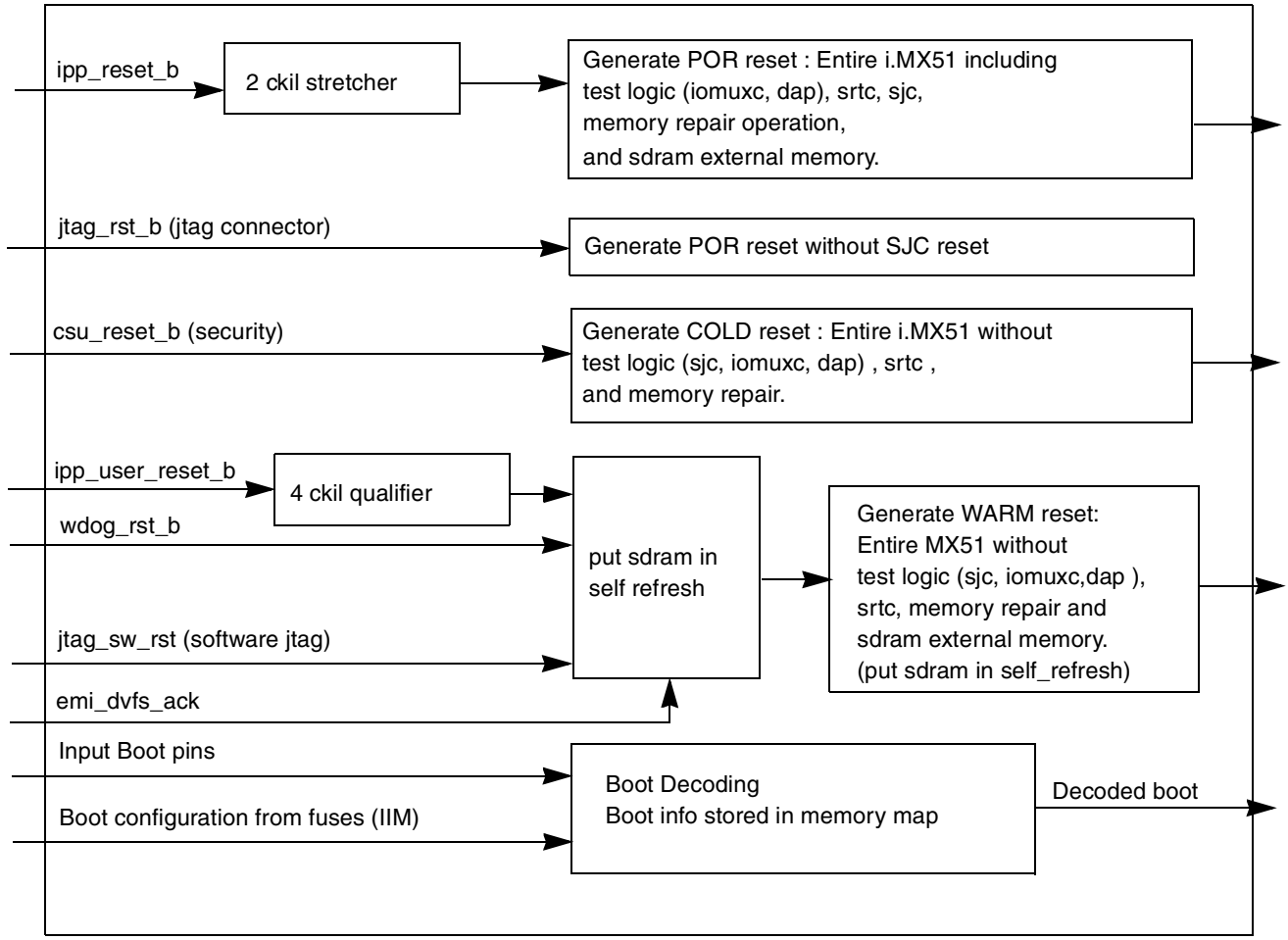


Figure 54-1. SRC High-Level Diagram

54.1.2 Features

The SRC includes the following features.

- Receives and handles the resets from all the reset sources
- Reset the appropriate domains based upon the resets sources and the nature of the reset.
- Latches the BMOD pins and common configuration signals from the internal fuse
- The SRC has 32-bit IP bus interface

54.2 Register Definition

This section contains a register summary and key to register conventions.

54.2.1 Register Summary

The conventions in Figure 54-2 and Table 54-1 serve as a key for the register summary and individual register diagrams.

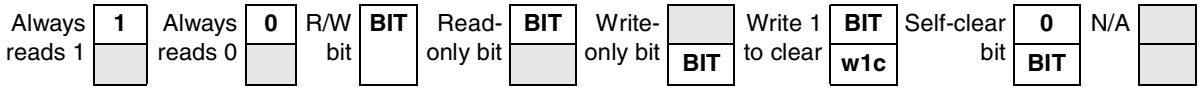


Figure 54-2. Key to Register Fields

Table 54-1 provides a key for register figures and tables and the register summary.

Table 54-1. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 54-2 shows the register summary for SRC.

Table 54-2. Sample Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBase_000 (SCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	mask_wdog_rst_b				warm_rst_bypass_count		sw_open_vg_rst	sw_ipu_rst	sw_vpu_rst	sw_gpu_rst	warm_reset_enable
	W																

Table 54-2. Sample Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBase_004 (SBMR)	R	BT_LPB_FREQ[2:0]			BT_HPN_EN	BT_USB_SRC	BT_UART_SRC[1:0]		BT_LPB[1:0]		BT_OSC_FREQ_SEL[1:0]		BT_SRC[1:0]		BT_LPB_EN	BT_WEIM_MUXED[1:0]	
	W																
	R	BMOD[1:0]		DIR_BT_DIS	BT_EEPROM_CFG		BT_MLC_SEL		BT_MEM_TYPE[1:0]		BT_SPARE_SIZE	0	BT_PAGE_SIZE[1:0]		BT_BUS_WIDTH	BT_MEM_CTL[1:0]	
	W																
0xBase_008 (SRSR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Warm_boot
	W																
	R										jtag_sw_rst	jtag_rst_b	wdog_rst_b	ipp_user_reset_b	csu_reset_b		ipp_reset_b
	W										w1c	w1c	w1c	w1c	w1c		w1c
0xBase_014 (SISR)	R																
	W																
	R													**open_vg_passed_reset**	**ipu_passed_reset**	**vpu_passed_reset**	**gpu_passed_reset**
	W																
0xBase_018 (SIMR)	R																
	W																
	R														**mask_ipu_passed_reset**		**mask_vpu_passed_reset**
	W																

54.2.2 Register Descriptions

This section consists of register descriptions in address order. All registers are reseted on POR sequence.

54.2.2.1 SRC Control Register (SCR)

Figure 54-3 presents the Reset control register (SCR), which contains bits that control operation of the reset controller. Table 54-3 provides its field descriptions.

0xBase_000 (SCR) Access: Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	mask_wdog_rst				warm_rst_bypass_count		sw_open_vg_rst	sw_ipu_rst	sw_vpu_rst	sw_gpu_rst	warm_reset_enable
W																
Reset	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1

Figure 54-3. SRC Control Register (SCR)

Table 54-3. SCR Field Descriptions

Field	Description
31–8	Reserved
10-7 mask_wdog_rst	Mask wdog_rst_b source. If these 4 bits are coded from A to 5 then, wdog_rst_b input to SRC will be masked and wdog_rst_b will not create reset to the IC. 0101 wdog_rst_b is masked 1010 wdog_rst_b is not masked (default) any other code will be coded to 1010 i.e. wdog_rst_b is not masked Note: During the time the WDOG event is masked using SRC logic, it is likely that WDOG Reset Status Register (WRSR) bit 1 (which indicates WDOG timeout event) will get asserted. SW / OS developer must prepare for this case. Re-enable WDOG is possible, by un-mask it in SRC, though it must be preceded by servicing the WDOG. However, for the case that the event has been asserted, the status bit (WRSR bit-1) will remain asserted, regardless of servicing the WDOG module. (HW reset is the only mean to cause de-assertion of that bit).
5-6 warm_rst_bypass_count	Defines the ckil cycles to count before bypassing the emi acknowledge for warm reset. If the emi acknowledge will not be asserted before this counter has elapsed, then a cold reset will be initiated. 00 Counter not to be used - system will wait until emi acknowledge until it is asserted. 01 Wait 16 ckil cycles before changing warm reset to a cold reset. 10 Wait 32 ckil cycles before changing warm reset to a cold reset. 11 Wait 64 ckil cycles before changing warm reset to a cold reset

Table 54-3. SCR Field Descriptions (continued)

Field	Description
4 sw_open_vg_rst	Software reset for open_vg 0 do not assert open_vg reset 1 assert open_vg reset Note: this is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrutp once the software has finished. Please referto SISR register for details. N ote: the reset process will involve 8 open_vg cycles before negating the open_vg reset, to allow reset assertion to propagate into open_vg.
3 sw_ipu_rst	Software reset for ipu 0 do not assert ipu reset 1 assert ipu reset Note: this is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrutp once the software has finished. Please referto SISR register for details.
2 sw_vpu_rst	Software reset for vpu 0 do not assert vpu reset 1 assert vpu reset Note: this is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrutp once the software has finished. Please referto SISR register for details. N ote: the reset process will involve 8 vpu cycles before negating the vpu reset, to allow reset assertion to propagate into vpu.
1 sw_gpu_rst	Software reset for gpu 0 do not assert gpu reset 1 assert gpu reset Note: this is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrutp once the software has finished. Please referto SISR register for details. N ote: the reset process will involve 8 gpu cycles before negating the gpu reset, to allow reset assertion to propagate into gpu.
0 warm_reset_enable	Warm reset enable bit. Warm reset will be enabled only if warm_reset_enable bit is set. Otherwise all warm reset sources will generate cold reset. 0 Warm reset disabled 1 Warm reset enabled

54.2.2.2 SRC Boot Mode Register (SBMR)

Figure 54-4 presents the Boot Mode register (SBMR), which contains bits that reflect the status of Boot Mode Pins of the chip. The default values for those bits depends on the values of pins/fuses during reset sequence, hence the question mark on their default value. Table 54-4 provides its field descriptions.

0xBase_004 (SBMR) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BT_LPB_FREQ[2:0]			BT_HPN_EN	BT_USB_SRC	BT_UART_SRC[1:0]		BT_LPB[1:0]		BT_OSC_FREQ_SEL[1:0]		BT_SRC[1:0]		BT_LPB_EN	BT_WEIM_MUXED[1:0]	
W																
Reset	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BMOD[1:0]		DIR_BT_DIS	BT_EEPROM_CFG		BT_MLC_SEL		BT_MEM_TYPE[1:0]		BT_SPARE_SIZE	0	BT_PAGE_SIZE[1:0]		BT_BUS_WIDTH	BT_MEM_CTL[1:0]	
W																
Reset	?	?	?	?	?	?	?	?	?	?	0	?	?	?	?	?

Figure 54-4. SRC Boot Mode Register (SBMR)

Table 54-4. SBMR Field Descriptions

Field	Description
31–29 BT_LPB_FREQ[2:0]	Please refer to fuse map.
28 BT_HPN_EN	Please refer to fuse map.
27 BT_USB_SRC	Please refer to fuse map.
26-25 BT_UART_SRC[1:0]	Please refer to fuse map.
24-23 BT_LPB[1:0]	Please refer to fuse map.
22-21 BT_OSC_FREQ_SEL[1:0]	Please refer to fuse map.
20-19 BT_SRC[1:0]	Please refer to fuse map.
18 BT_LPB_EN	Please refer to fuse map.

Table 54-4. SBMR Field Descriptions (continued)

Field	Description
17-16 BT_WEIM_MUXED[1:0]	Please refer to fuse map.
15 – 14 BMOD[1:0]]	Please refer to fuse map.
13 DIR_BT_DIS	Please refer to fuse map.
12 BT_EEPROM_CFG	Please refer to fuse map.
11	
10 BT_MLC_SEL	Please refer to fuse map.
9	Reserved.
8 - 7 BT_MEM_TYPE[1:0]	Please refer to fuse map.
6 BT_SPARE_SIZE	Please refer to fuse map.
5	Reserved.
4 - 3 BT_PAGE_SIZE[1:0]	Please refer to fuse map.
2 BT_BUS_WIDTH	Please refer to fuse map.
1 - 0 BT_MEM_CTL[1:0]	Please refer to fuse map.

54.2.2.3 SRC Reset Status Register (SRSR)

Figure 54-5 presents the SRC Reset Status register of the chip. The SRSR is a write to one clear register which records the source of the reset events for the chip. The SRC reset status register will capture all the reset sources that have occurred. Table 54-5 provides its field descriptions.

0xBase_008 (SRSR) Access: Read/Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Warm_boot
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R										jtag_sw_rst	jtag_rst_b	wdog_rst_b	ipp_user_reset_b	csu_reset_b		ipp_reset_b
W										w1c	w1c	w1c	w1c	w1c		w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 54-5. SRC Reset Status Register (SRSR)

This register is reset on $\overline{\text{ipp\_reset\_b}}$. This is a read-write register,

For bit[6-0] - writing zero does not have any effect. Writing one will clear the corresponding bit. The individual bits can be cleared by writing one to that bit. When the system comes out of reset, this register will have bits set corresponding to all the reset sources that occurred during system reset. Software has to take care to clear this register by writing one after every reset that occurs so that the register will contain the information of recently occurred reset.

Table 54-5. SRSR Field Descriptions

Field	Description
31–17	Reserved
16 warm_boot	Warm boot indication gives software capability to notify that warm boot was initiated by software. This indicates to the software that it saved the needed info in the memory before initiating the warm reset. In this case , software will set this bit to '1', before initiating the warm reset. The warm_boot bit should be used as indication only after a warm_reset sequence. Software should clear this bit after warm_reset to indicate that next warm_reset is not done with warm_boot. Please refer to Section 54.3.1.2.2, Reset Sequence and De-Assertion for details on warm_reset. 0 warm boot process not initiated by software. 1 warm boot initiated by software.
15–7	Reserved

Table 54-5. SRSR Field Descriptions (continued)

Field	Description
6 jtag_sw_rst	JTAG SW reset. Indicates whether the reset was the result of software reset from JTAG. 0 Reset is not a result of software reset from JTAG. 1 Reset is a result of software reset from JTAG. Connections at chip-level: jtag_sw_rst -> sjc_gpccr_reg_2_b
5 jtag_rst_b	HIGH - Z JTAG reset. Indicates whether the reset was the result of HIGH-Z reset reset from JTAG. 0 Reset is not a result of HIGH-Z reset from JTAG. 1 Reset is a result of HIGH-Z reset from JTAG. Connections at chip-level: jtag_rst_b -> sjc_ieee_reset_b
4 wdog_rst_b	IC Watchdog Time-out reset. Indicates whether the reset was the result of the watchdog time-out event. 0 Reset is not a result of the watchdog time-out event. 1 Reset is a result of the watchdog time-out event.
3 ipp_user_reset_b	Indicates whether the reset was the result of the ipp_user_reset_b qulified reset. 0 Reset is not a result of the ipp_user_reset_b qulified as COLD event. 1 Reset is a result of the ipp_user_reset_b qulified as COLD event.
2 csu_reset_b	Indicates whether the reset was the result of the csu_reset_b input. 0 Reset is not a result of the csu_reset_b event. 1 Reset is a result of the csu_reset_b event. Note: If case the csu_reset_b occured during a warm reset process, during the phase that ipg_clk is not available yet, then the occurance of csu reset will not be reflected in this bit.
1	Reserved.
0 ipp_reset_b	Indicates whether reset was the result of ipp_reset_b pin (Power-up sequence) 0 Reset is not a result of ipp_reset_b pin. 1 Reset is a result of ipp_reset_b pin.

54.2.2.4 SRC Interrupt Status Register(SISR)

Figure 54-8 presents the SRC Interrupt Status Register (SISR). Table 54-8 provides its field descriptions.

0xBase_014 (SISR) Access: User write one to clear

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R													open_vg_passed_reset	ipu_passed_reset	vpu_passed_reset	gpu_passed_reset
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 54-8. SRC Interrupt Status Register (SISR)

Table 54-8. SISR Field Descriptions

Field	Description
31-4	Reserved
3 open_vg_passed_reset	Interrupt generated to indicate that open_vg passed software reset and is ready to be used 0 - interrupt genrerated not due to open_vg passed reset 1 - interrupt genrerated due to open_vg passed reset
2 ipu_passed_reset	Interrupt generated to indicate that ipu passed software reset and is ready to be used 0 - interrupt genrerated not due to ipu passed reset 1 - interrupt genrerated due to ipu passed reset
1 vpu_passed_reset	Interrupt generated to indicate that vpu passed software reset and is ready to be used 0 - interrupt genrerated not due to vpu passed reset 1 - interrupt genrerated due to vpu passed reset
0 gpu_passed_reset	Interrupt generated to indicate that gpu passed software reset and is ready to be used 0 - interrupt genrerated not due to gpu passed reset 1 - interrupt genrerated due to gpu passed reset

54.2.2.5 SRC Interrupt Mask Register(SIMR)

Figure 54-9 presents the SRC Interrupt Mask Register (SIMR). Table 54-9 provides its field descriptions.

0xBase_018 (SIMR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R													mask_open_vg_passed_reset	mask_ipu_passed_reset	mask_vpu_passed_reset	mask_gpu_passed_reset
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Figure 54-9. SRC Interrupt Mask Register (SIMR)

Table 54-9. SIMR Field Descriptions

Field	Description
31-4	Reserved
3 mask_open_vg_passed_reset	mask interrupt generation due to open_vg passed reset 0 - don't mask interrupt due to open_vg passed reset - interrupt will be created 1 - mask interrupt due to open_vg passed reset
2 mask_ipu_passed_reset	mask interrupt generation due to ipu passed reset 0 - don't mask interrupt due to ipu passed reset - interrupt will be created 1 - mask interrupt due to ipu passed reset
1 mask_vpu_passed_reset	mask interrupt generation due to vpu passed reset 0 - don't mask interrupt due to vpu passed reset - interrupt will be created 1 - mask interrupt due to vpu passed reset
0 mask_gpu_passed_reset	mask interrupt generation due to gpu passed reset 0 - don't mask interrupt due to gpu passed reset - interrupt will be created 1 - mask interrupt due to gpu passed reset

54.3 Functional Description

54.3.1 Reset Control

This section details the reset control of this device.

54.3.1.1 Reset Inputs and Outputs

The Reset control logic receives reset requests from all the potential reset sources. All the immediate sources of reset are directly passed to the reset stretching block whereas the resets which require qualification are passed on to the reset qualification logic before they are sent to the reset stretching block.

All reset inputs and outputs are described in Figure 54-10:

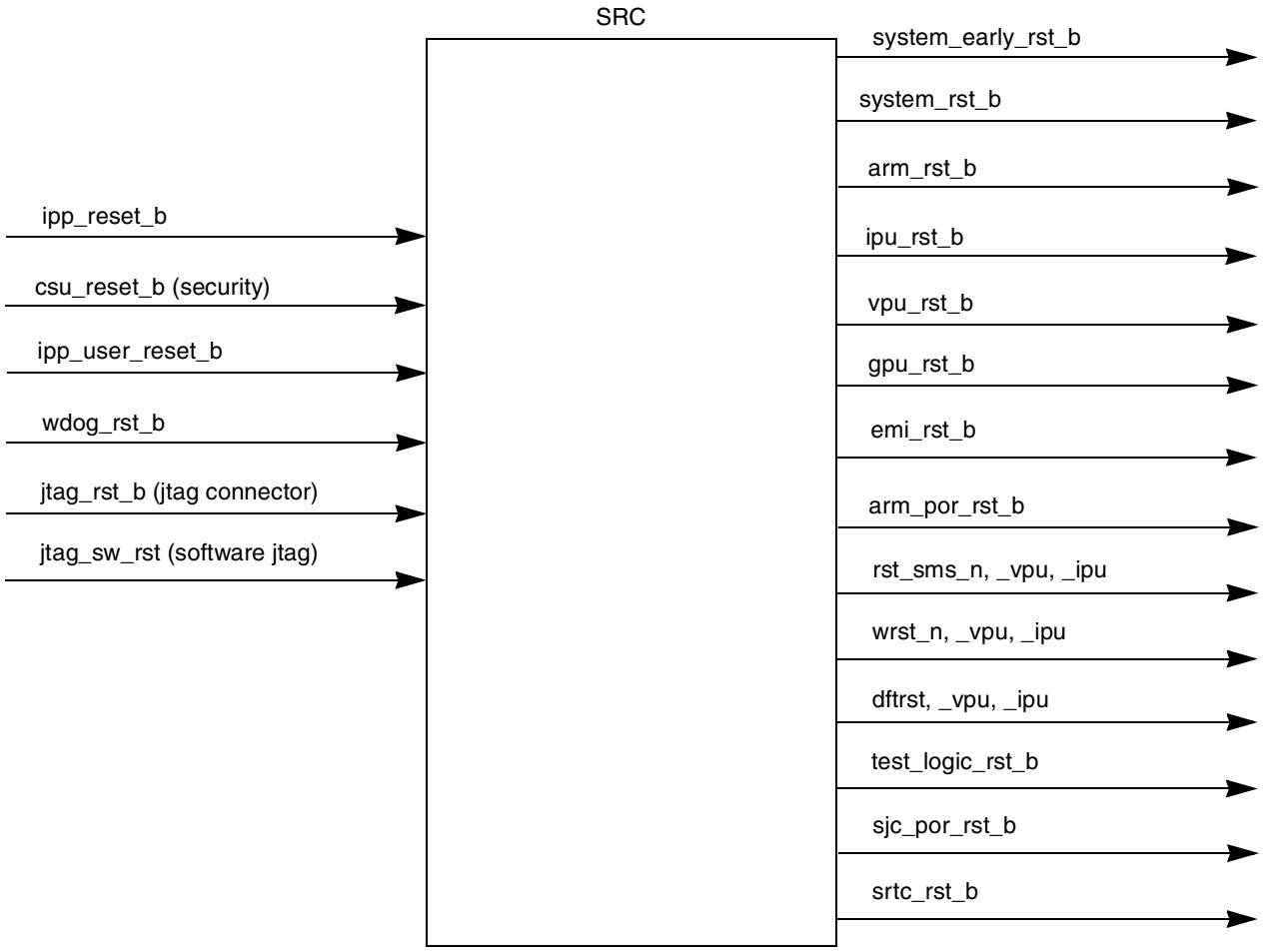


Figure 54-10. SRC Inputs and Outputs

The reset sources, type and behavior are shown in the Table 54-10. As there is no chip POR the $\overline{\text{ipp\_reset\_b}}$ is used to reset the entire chip including test logic and JTAG modules.

NOTE

All resets are expected to be active low except jtag_sw_rst.

Table 54-10. Reset Priorities, Sources, Types, and Behavior

Priority	Sources	Type	Behavior
1	ipp_reset_b	POR	Reset IC = system_early_rst_b + system_rst_b + arm_por_rst+ arm_soc_rst_b + arm_dbg_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b + open_vg_rst_b + emi_rst_b + rst_sms_n + wrst + dftrst + test_logic_rst_b + srtc_rst_b + sjc_por_rst_b
2	csu_reset_b	Cold	Reset IC - rst_sms_n - wrst - dftrst - test_logic_rst_b - srtc_rst_b - sjc_por_rst_b - arm_por_rst - arm_dbg_rst_b = system_early_rst_b + system_rst_b + arm_soc_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b + open_vg_rst_b + emi_rst_b
3	ipp_user_reset_b (qualified 4 ckil's)	Warm	Reset IC - rst_sms_n - wrst - dftrst - test_logic_rst_b - srtc_rst_b- arm_por_rst - arm_dbg_rst_b - sjc_por_rst_b + warm_reset_signal = system_early_rst_b + system_rst_b + arm_soc_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b + open_vg_rst_b + emi_rst_b + warm_reset_signal
4	wdog_rst_b This reset will not be generated if mask_wdog_rst bits are coded to 5.	Warm	Reset IC - rst_sms_n - wrst - dftrst - test_logic_rst_b - srtc_rst_b - arm_por_rst - arm_dbg_rst_b - sjc_por_rst_b + warm_reset_signal = system_early_rst_b + system_rst_b + arm_soc_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b + open_vg_rst_b + emi_rst_b + warm_reset_signal
5	jtag_rst_b (jtag connector)	POR - sjc_por_rst_b	Reset IC - sjc_por_rst_b = system_early_rst_b + system_rst_b + arm_por_rst+ arm_soc_rst_b + arm_dbg_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b + open_vg_rst_b + emi_rst_b + rst_sms_n + wrst + dftrst + test_logic_rst_b + srtc_rst_b
6	jtag_sw_rst (software jtag)	Warm	Reset IC - rst_sms_n - wrst - dftrst - test_logic_rst_b - srtc_rst_b - arm_por_rst - arm_dbg_rst_b + warm_reset_signal - sjc_por_rst_b = system_early_rst_b + system_rst_b + arm_rst_b + ipu_rst_b + vpu_rst_b + gpu_rst_b+ open_vg_rst_b + emi_rst_b + warm_reset_signal

The reset priorities are POR (strongest), COLD and WARM (weakest). If a stronger reset is asserted during the sequence of a wicker reset, then the wicker sequence will be interfered and the stronger reset sequence will commence. There is no priority inside a reset type group. If a reset is asserted during reset sequence of reset from the same type group , then the reset sequence will be interfered and a new reset sequence (from the same type) will commence.

The following lists the functionality of each of these reset outputs:

- system_early_rst_b — Resets the system modules that need to start first as ccm, pll_ip's, iim, fuseboxes, emi etc.
- system_rst_b — Resets functional modules.
- arm_rst_b — Resets ARM module (on regular system reset)
- ipu_rst_b — Resets IPU module (on regular system reset and on power gating of ipu)
- vpu_rst_b — Resets VPU module (on regular system reset and on power gating of vpu)
- gpu_rst_b — Resets GPU module (on regular system reset and on power gating of gpu)
- open_vg_rst_b — Resets Open_VG module (on regular system reset and on power gating of open_vg)

- emi_rst_b — Resets SDRAM controller.
- arm_por_rst — Resets ARM por input.
- arm_soc_rst_b - Reset for arm SOC.
- arm_dbg_rst_b - Reset debug logic of ARM.
- test_logic_rst_b - Reset test logic. (iomuxc, dap)
- sjc_por_rst_b - reset to SJC.
- srtc_rst_b - Resets SRTC.

The following are related to SMS(STAR memory system):

- rst_sms_n - Resets SMS
- dftrst - Resets SMS DFT
- wrst - Resets JPC wrapper.

SRC will also generate rst_sms_n dftrst and wrst_n for specific accelerators. Since those modules are not involved in the memory repair process, then they will be similar to the functional resets of those modules. Note that dftrst is active high, hence its inverted version equalls the functional reset.

- dftrst_ipu = wrst_n_ipu = rst_sms_n_ipu = ipu_rst_b
- dftrst_vpu = wrst_n_vpu = rst_sms_n_vpu = vpu_rst_b

Note: It is assumed that each reset source will deassert after its assertion, either due to reset generated to the system from SRC, or either by negation of the reset source (in case it came from an external source to the IC). In the later case, the reset source is assumed to be held for atleast 2 ckil cycles so that SRC can sample it.

54.3.1.2 Reset Handling

54.3.1.2.1 Reset Qualification

The reset qualification logic qualifies the reset source before sending it out to the chip as a valid reset. Warm resets are in this category. All remaining reset sources are immediate resets and are acknowledged by the reset circuitry the moment they are asserted.

Warm resets are not immediate resets. Warm resets do reset the CCM, the source of SDRAMC clock. So, if a Warm reset were to immediately reset the CCM, then the SDRAMC clock would be shut off and this may cause the SDRAM data to be lost. During normal mode of operation of the chip, the protocol that is followed before shutting off the SDRAMC clock is that an emi_dvfs_req signal is sent to SDRAMC and only after the SDRAMC sends an acknowledge signal, emi_dvfs_ack, are the clock to the SDRAMC gated off.

However, the implication here is that a valid warm reset source condition will not be able to cause a chip reset until the SDRAMC sends the acknowledge signal (emi_dvfs_ack). For example, a JTAG reset event has occurred but the JTAG reset will not be serviced until the emi_dvfs_ack signal is received. So, essentially all Warm reset sources depend on the SDRAMC providing the emi_dvfs_ack acknowledge signal before the reset is performed. When the SDRAM controller is not used, emi_dvfs_ack is defaulted high.

The occurrence of warm reset results in the assertion of warm_reset signal before the system resets are asserted to indicate EMI that the reset occurred is a warm reset.

A reset source is updated in the Reset status register when it becomes valid, provided it is asserted for the minimum amount of time after asserting. So, all immediate resets would be immediately updated in the Status register. Warm resets would be updated when the emi_dvfs_ack signal is received from the SDRAMC.

Once the reset is qualified, depending on the source of the reset, internal resets would be asserted appropriately.

54.3.1.2.2 Reset Sequence and De-Assertion

The system_early_rst_b and the system_rst_b will assert immediately after any reset source is recognized (except for the case of warm reset when emi needs to answer emi_dvfs_ack first). After all reset sources are released, the SRC initiates the following set of events depending on the type of reset that occurred.

$\overline{\text{IPP\_RESET\_B}}$(POR)

This is a external signal from the pads and whenever the chip is powered up the reset signal is passed through this pin indicating power-up sequence and the SRC resets the entire chip including test logic and JTAG module. All SRC registers will be reset on the POR sequence, as follows.

1. As soon as $\overline{\text{IPP\_RESET\_B}}$ occurs all resets are asserted and the entire chip is reset by SRC Module. The $\overline{\text{IPP\_RESET\_B}}$ is stretched for 2 CKIL cycles and the stretching sequence takes place after 2 CKIL clocks of $\overline{\text{IPP\_RESET\_B}}$ pin de-assertion.
2. test_logic_rst_b, sjc_por_rst_b and srtc_rst_b signals are de-asserted together with IPP_RESET_B signal. Those outputs are also deasserted after the stretching of IPP_RESET_B has deasserted.
3. After the above resets de-asserts, system_early_rst_b reset is de-asserted after 2 CKIL clock. The system_early_rst_b is used for the CCM and PLL-IP's to start generating pll clock ouputs and the system root clocks.
4. Once system root clocks are ready, CCM will assert system_clk_ready signal. This signal is generated during the ignition sequence in CCM and it involves the preparation of FPM/CKIH/PLL's to generate clock roots for functional operation. Please refer to Ignition Sequence in CCM chapter.
5. Once this signal arives to SRC , it will initiate smart sfp operation. This operation will be finished once the SRC receives the assertion on signal "sfp_ready".
6. After SFP operation finish, SRC will enable IIM and fusebox clocks, so that fuses can be loaded to IIM. IIM will notify with iim_ready_flag on the finish of the fusebox loading.
7. SRC will prepare the boot information and then de assert emi_rst_b.
8. SRC will wait 8 ipg clock cycles, and then SRC will enable EMI clocks.
9. SRC will wait 8 CKIL clocks to allow EMI to generate fixed external clock to external memory SDRAM.
10. SRC will enable VPU and GPU clocks so that reset will penetrate this module.
11. After 8 ipg cycles, src will dissable vpu, open_vg and gpu clocks.

12. After 8 ipg cycles, resets to all modules will be de-asserted (system_rst_b, vpu_rst_b, gpu_rst_b, open_vg_rst_b, ipu_rst_b).
13. If en_tester_control signal is asserted, SRC will wait for assertion of tester_ack signal to continue. This signal allows tester to receive determinism on the reset sequence.
14. After 8 ipg cycles, system clocks will be enabled (en_system_clk).

NOTE

The memory repair operation is bypassed when the smart_bypass signal is asserted.

COLD RESET

The sequence is similar to $\overline{\text{IPP\_RESET\_B}}$ except the memory repair operation is not performed and test_logic_reset is not fired.

1. Once the reset source de-asserts, system_early_rst_b reset is de-asserted after at least 2 CKIL clock. The system_early_rst_b is used for the CCM and PLL-IP's to start generating pll clock ouputs and the system root clocks.
2. Once system root clocks are ready, CCM will assert system_clk_ready signal. This signal is generated during the ignition sequence in CCM and it involves the preparation of FPM/CKIH/PLL's to generate clock roots for functional operation. Please refer to Ignition Sequence in CCM chapter.
3. Once this signal arives to SRC ,SRC will enable IIM and fusebox clocks, so that fuses can be loaded to IIM. IIM will notify with iim_ready_flag on the finish of the fusebox loading.
4. SRC will prepare the boot information and then de assert emi_rst_b.
5. SRC will wait 8 ipg clock cycles, and then SRC will enable EMI clocks.
6. SRC will wait 8 CKIL clocks to allow EMI to generate fixed external clock to external memory SDRAM.
7. SRC will enable VPU and GPU clocks so that reset will penetrate this module.
8. After 8 ipg cycles, src will dissable vpu, open_vg and gpu clocks.
9. After 8 ipg cycles resets to all modules will be de-asserted (system_rst_b, vpu_rst_b, gpu_rst_b, open_vg_rst_b, ipu_rst_b).
10. If en_tester_control signal is asserted, SRC will wait for assertion of tester_ack signal to continue. This signal allows tester to receive determinism on the reset sequence.
11. After 8 ipg cycles, system clocks will be enabled (en_system_clk).

WARM RESET

WARM reset will be enabled only if warm_reset_enable bit is programed. Otherwise all warm reset sources will generate cold reset. This bit will be reset only by por reset.

A WARM reset is similar to a cold reset except that before the reset is fired a signal to emi is asserted emi_dvfs_req (generates dvfs assertion to emi) to request to prepare emi to a WARM reset, i.e. to finish the transactions and to put the sdram in selfrefresh. Another signal will be asserted to emi (warm_reset) that will wrap the warm reset sequence and will notify emi that a warm reset is in process.

One of the sources of the WARM reset is ipp_user_reset_b. If this is the case, it is qualified for 4 CKIL edges. The WARM reset is initiated immediately after 4 CKIL edges. The system does not come out of reset until the the ipp_user_reset_b is released.

During WARM reset, request to put SDRAM in self refresh is sent to SDRAMC. The request is sent through generation of dvfs signal to EMI. The request is passed through CCM to be combined with CCM's capability to request dvfs. Only after the EMI sends an acknowledge signal, emi_dvfs_ack is warm reset generated. The implication here is that a valid WARM reset source condition will not be able to cause a chip reset until the EMI sends the acknowledge signal (emi_dvfs_ack). For example, a WDOG reset event has occurred but the WDOG reset will not be serviced until the emi_dvfs_ack signal is received. So, essentially all WARM reset sources depend on the EMI providing the emi_dvfs_ack acknowledge signal before the reset is performed. When the EMI is not used, emi_dvfs_ack is defaulted high.

After the acknowledge signal, emi_dvfs_ack , is received the warm reset is generated. The warm_reset input to SDRAMC reflects the value of the warm_reset_enable bit. This signal indicates to SDRAMC that the upcoming reset is WARM.

In case the handshake mechanism with SDRAMC is stuck meaning no emi_dvfs_ack is recieved, cold reset will be generated after a number of CKIL clocks. The number of ckil clocks is defined in register SCR bit warm_rst_bypass_count. the default of this bit is 16 ckil count.

The following is a basic description of the warm reset sequence:

1. ARM sets warm_reset_enable bit to enable the WARM Reset functionality. If this bit is not set, all WARM reset sources will result in Cold Reset.
2. Assertion of one of the WARM reset sources.
3. The reset source is qualified in the SRC.
4. If emi_dvfs_ack signal is low, then SRC triggers the SDRAM Ctrl. to switch to self refresh mode using emi_dvfs_req signal. This is done threw CCM to combine with the dvfs sent from CCM in case of frequency change of EMI.
5. Wait for emi_dvfs_ack signal from the EMI. If no ack is recieved during warm_rst_bypass_count number of CKIL clocks, cold reset will be generated.
6. Assert warm_reset signal to EMI.
7. SRC asserts system resets
8. The de-assertion sequence is exactly the same as in the Cold Reset except waiting for 8 CKIL's for EMI to generate fixed external clock to external memory SDRAM. This stage is not needed in WARM reset since SDRAM is held in selfrefresh in WARM reset and there is no need to reconfigure it when going out of WARM reset.

WARM BOOT

Software can save any needed information in the memory before initiating a warm reset. In this case, software will set warm_boot bit before initiating warm reset. After the system returns to run mode, the warm_boot bit will still be set, indicating the software that data was saved in memory and can be reused.

Figure 54-11, Figure 54-12, Figure 54-13, Figure 54-14 shows the behavior of different resets for POR, Cold, and Warm reset cases respectively.

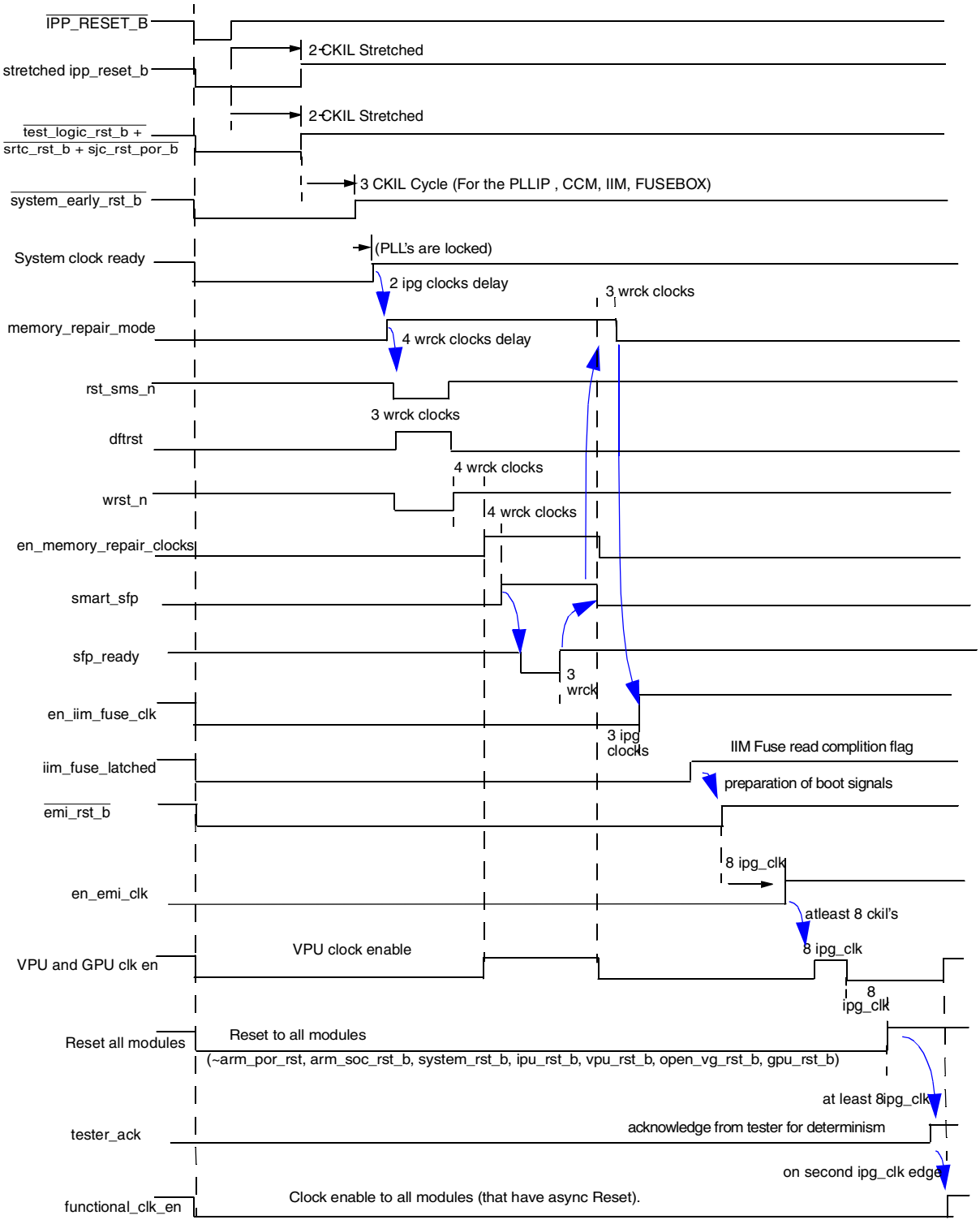


Figure 54-11. Behavior of IPP_RESET_B(POR)

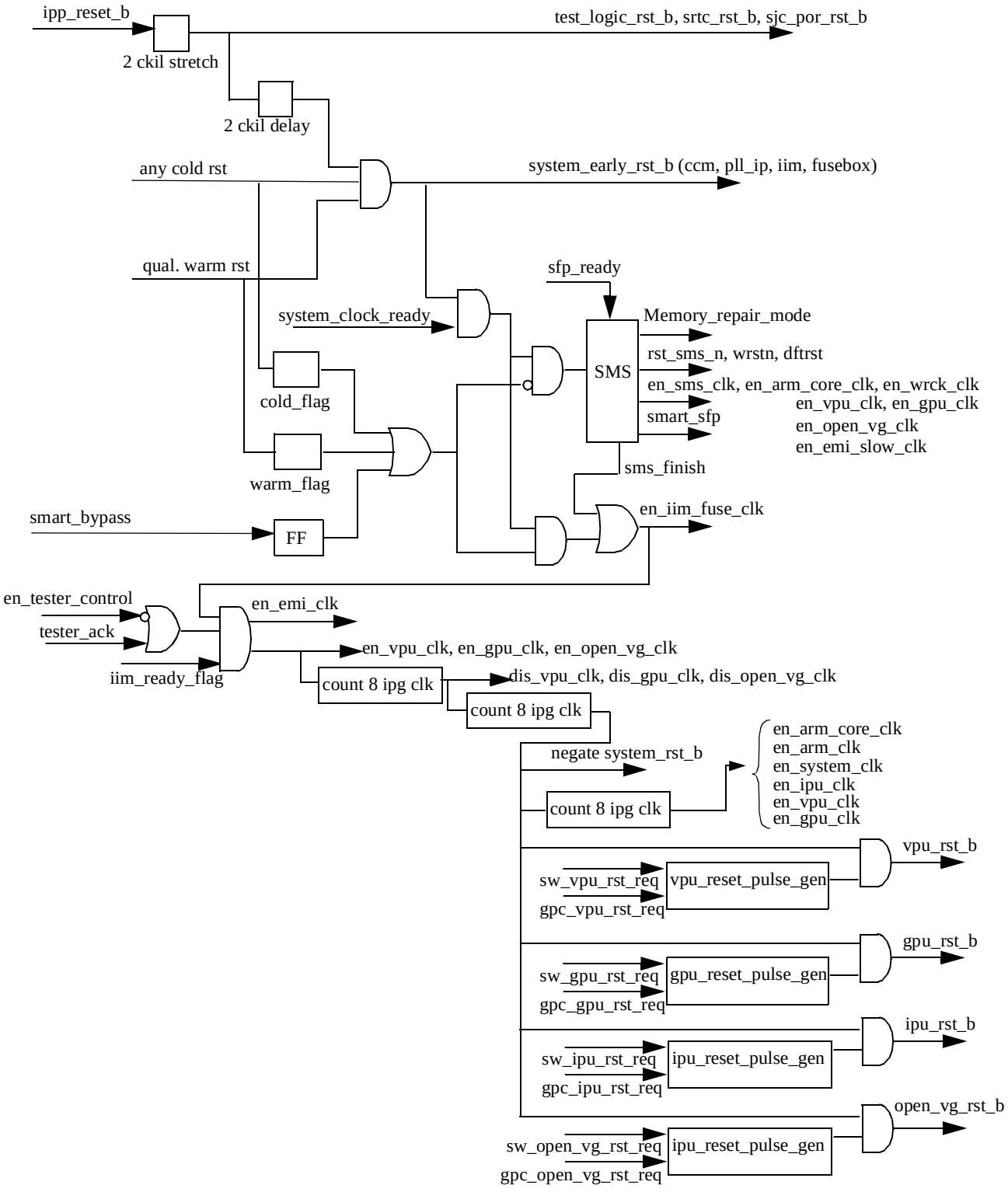


Figure 54-12. Reset Diagram

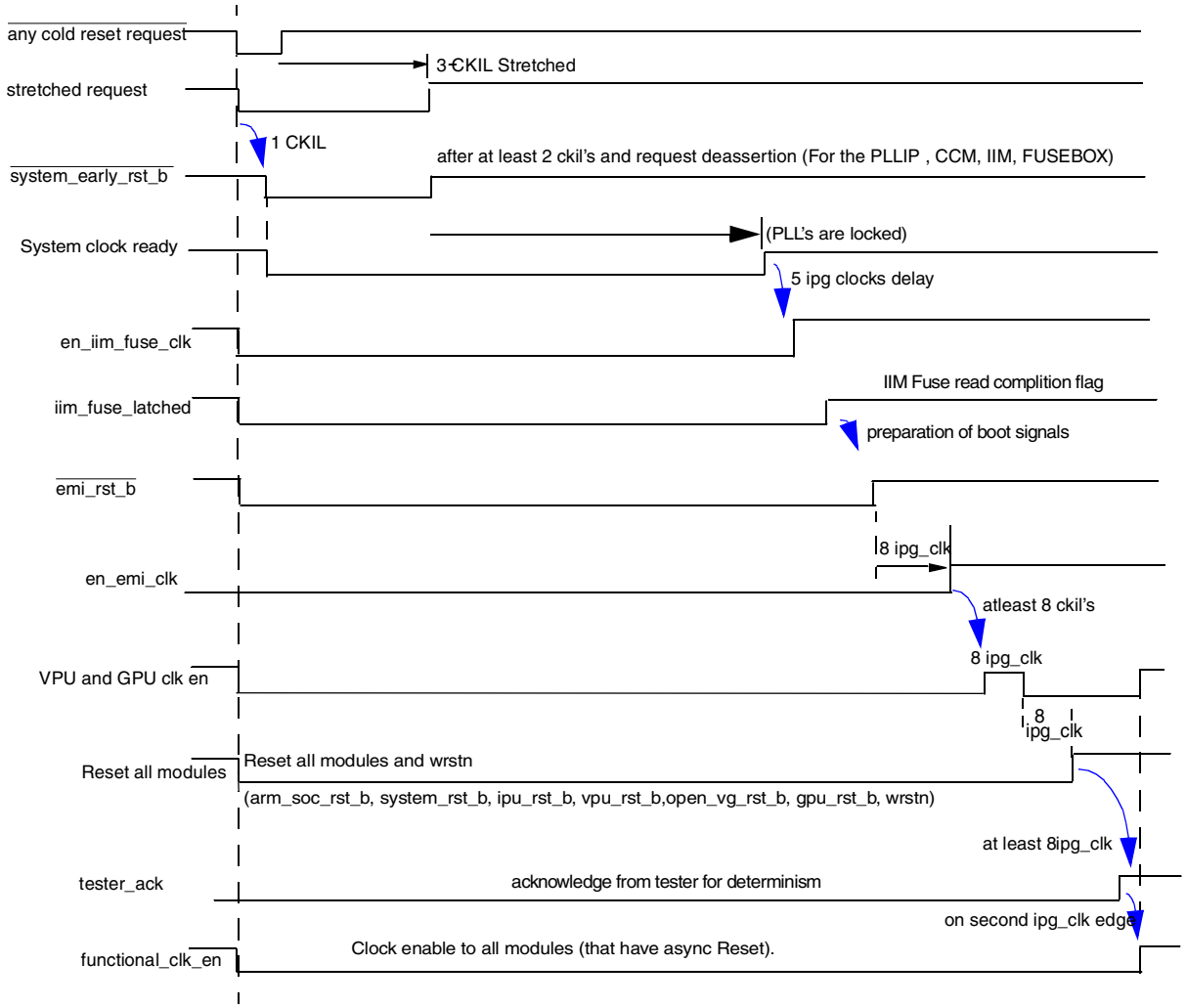


Figure 54-13. Cold Reset Sequence

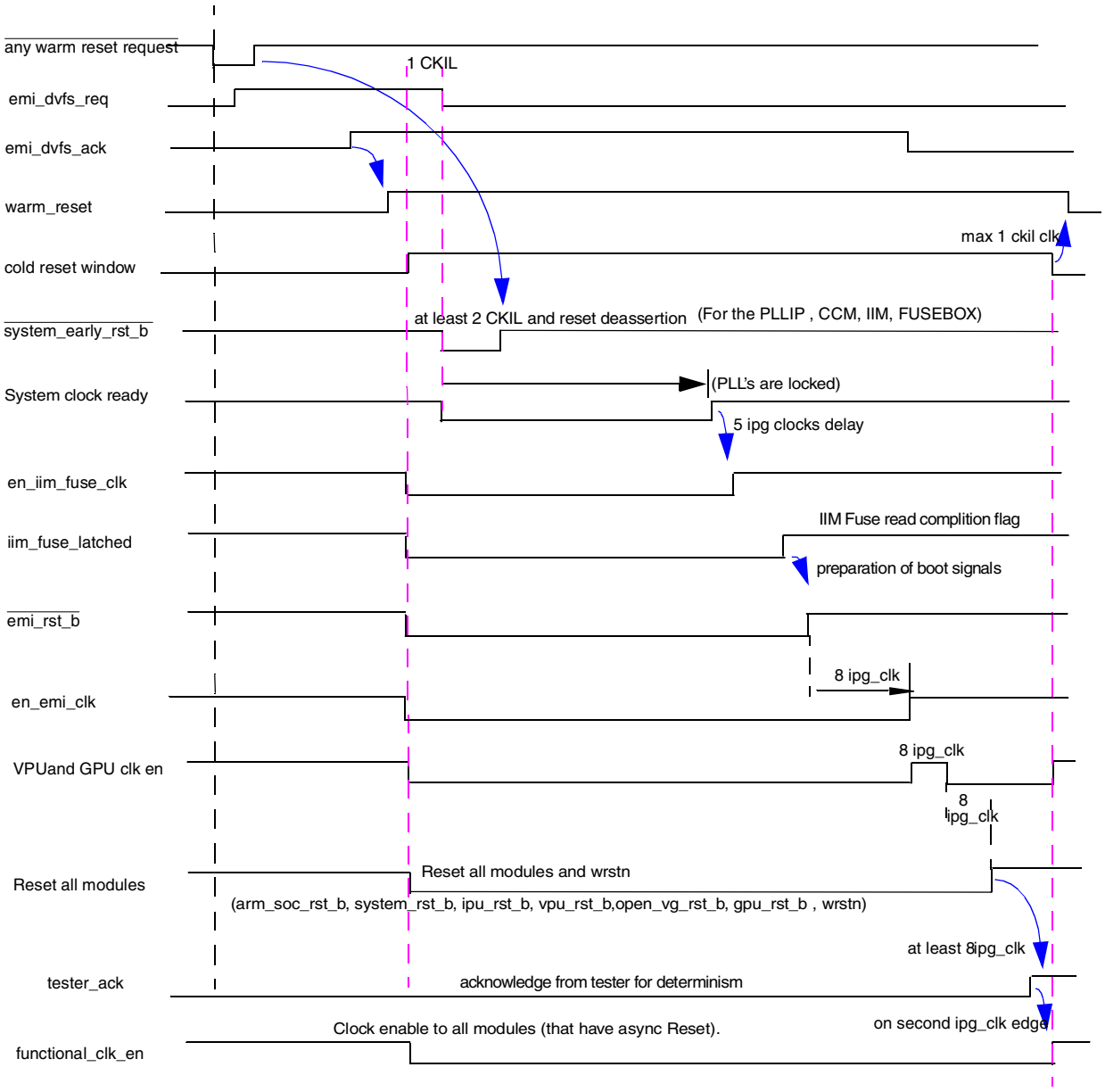


Figure 54-14. Warm Reset Sequence

54.3.1.2.3 SMS Submodule

The sms submodule manages the memory repair signals during por. The sms sub module will start its operation after system_clock_ready notification from CCM and only if the reset sequence is due to por sequence (ipp_reset_b). The SMS submodule operation can be bypassed by smart_bypass assertion signal from the pad. If the smart_bypass is asserted, only the assertion part of sms reset signals will take place, but smart_sfp - sfp_ready sequence will not be perofrmed.

The SMS sequence is as follows:

1. Upon assertion of system_clock_ready signal, count 2 ipg clocks and assert the memory_repair_mode signal.
2. Wait 4 wrck clocks.
3. Assert sms reset signals on wrck clock edge:
   - rst_sms_n = 0
   - wrst_n = 0
   - dftrst = 1
4. Wait 3 wrck clocks
5. . De assert the reset signals on wrck clock edge:

- rst_sms_n = 1
- wrst_n = 1
- dftrst = 0

6. Wait 4 wrck clocks
7. Enable memory repair clocks, i.e. en_arm_core_clk=1, en_emi_slow_clk=1, en_vpu_clk=1, en_gpu_clk=1, en_open_vg_clk=1, en_sms_clk=1 and en_wrck_clk=1 on wrck edge.
8. Wait 4 wrck clocks
9. Assert smart_sfp signal on wrck edge.
10. Monitor de assertion and assertion of sfp_ready signal.
11. Once assertion of sfp_ready signal, de assert smart_sfp, de assert clock enables en_arm_core_clk=0 , en_emi_slow_clk=0, en_vpu_clk=0, en_gpu_clk=0, en_open_vg_clk=0, en_sms_clk=0 and en_wrck_clk=0 on wrck edge, de assert memory_repair_mode signal. (all on wrck edges).
12. Wait 4 wrck clocks
13. Assert sms_finish to notify that SRC can continue with enable IIM clocks .

Figure 54-15 describes the SMS flow.

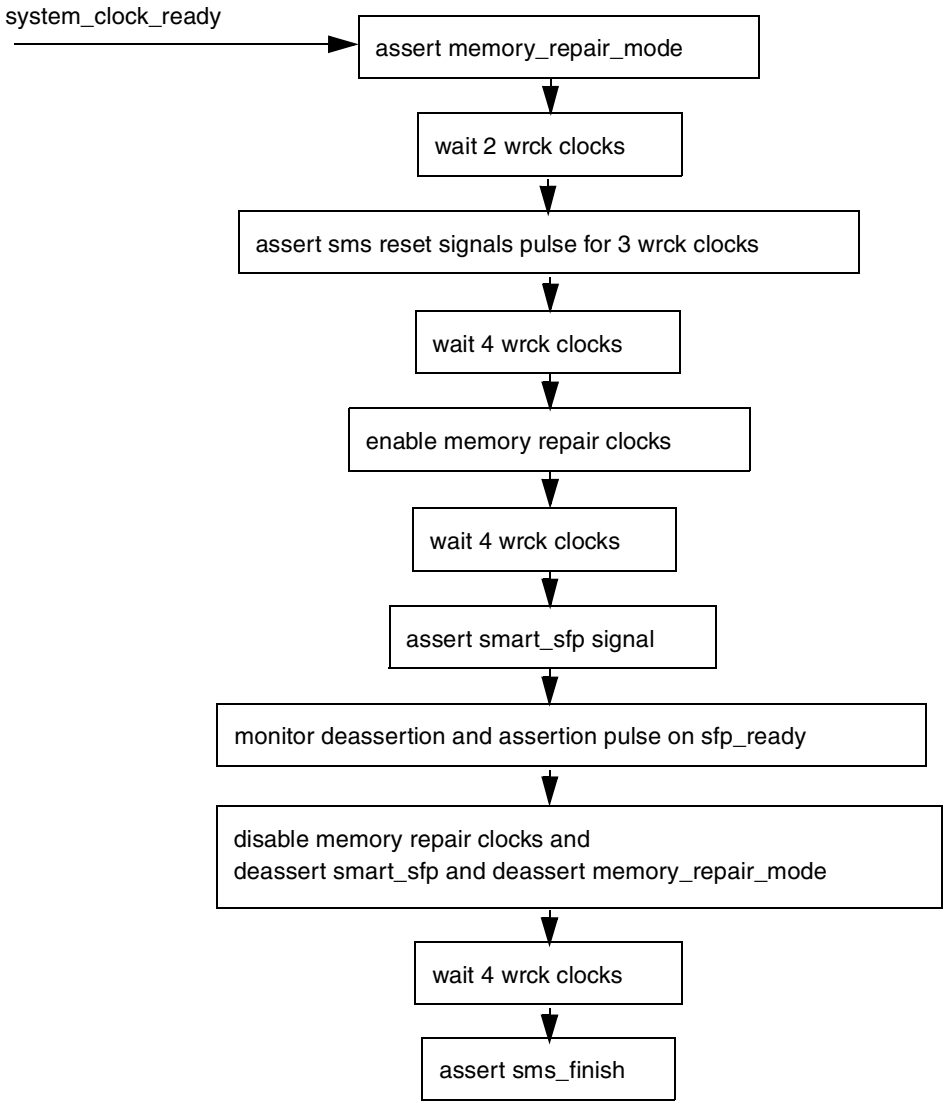


Figure 54-15. SMS flow diagram

54.3.1.2.4 Reset Pulse Generators

Except from POR, cold, and warm sequence resets, a reset pulse can be generated for VPU, Open VG, GPU and IPU. Those reset pulses can be generated upon sw reset bit assertion, or by signal from GPC requesting the reset pulse. Once the sw bit or the gpc signal are asserted, the respective reset will be asserted.

If the reset source is GPC request , then once the GPC signal is deasserted, the respective reset will be deaseerted.

If the reset source is sw bit, then once the reset asertion sequence will be finished, the reset will be deasserted, and after reset deassertion sequence , the sw bit will be cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software reset has finished. Please refer to SISR register for details.

Below is explanation how each of those reset pulse generators are functioning:

IPU reset pulse generator:

Upon assertion of sw_ipu_rst bit (sw_ipu_rst = ‘1’) or assertion of gpc_reset_ipu_req (gpc_reset_ipu_req=’0’) then SRC should do the following:

1. If the request came from gpc, i.e. gpc_reset_ipu_req=’0’ then negate ipu_reset_done internally to SRC. This will cause negation of power_gating_reset_done signal to CCM.
2. Close ipu clocks (en_ipu_clk=’0’)
3. Assert ipu reset (ipu_rst_b=’0’)

SRC will continue with the reset deassertion step described below in the following cases:

- If the reset source was the software bit, then SRC will continue to the deassertion steps imediately.
- If the reset source was GPC request, then upon de assertion of gpc_reset_ipu_req (gpc_reset_ipu_req=’1’) SRC will continue to the deassertion steps:
  a) Deassert ipu reset (ipu_rst_b=’1’)
  b) Count at least 8 ipg_clk to allow reset to penetrate reset tree
  c) Open ipu clocks (en_ipu_clk=’1’)
  d) If the request came from gpc, i.e. gpc_reset_ipu_req caused the reset request then assert ipu_reset_done internally to SRC. If the request came from sw_ipu_rst bit assertion then generate interrupt (if its not masked) to notify that reset ipu passed reset and is ready to be used, and clear the sw_ipu_rst bit.

Figure 54-16 describes the flow of ipu reset pulse generator.

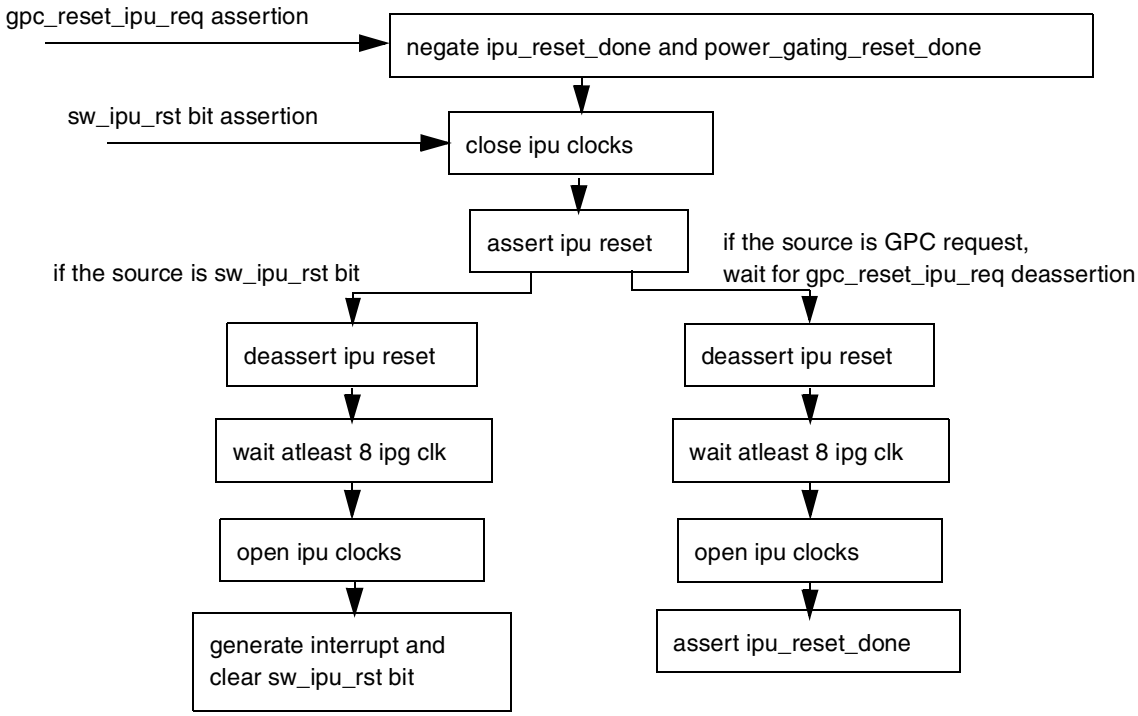


Figure 54-16. ipu Reset Pulse Generator

VPU reset pulse generator:

VPU reset pulse can be generated from sw_vpu_rst bit or gpc_reset_vpu_req.

If the request came from gpc, i.e. gpc_reset_vpu_req='0' then negate vpu_reset_done internally to SRC. This will cause negation of power_gating_reset_done signal to CCM. On this stage also deassert vpu_reset_penetrated.

1. Close vpu clocks (en_vpu_clk='0') (just to make sure they are closed)
2. Assert vpu reset (vpu_rst_b='0').
3. If the source was GPC request, wait for change from 1 to 0 on gpc_vpu_switch_b signal to notify that power is back.
4. Count atleast 8 ipg_clk clocks so that reset will propagate the reset tree
5. Open vpu clocks.
6. Count at least 16 ipg_clk clocks so that reset will propagate in vpu (this stage will happen only after pll relock and clocks are ready prior to exiting from stop)
7. Close vpu clocks (en_vpu_clk='0')
8. Wait at least 8 ipg_clk cycles so that clocks to vpu will be closed, and then assert vpu_reset_penetrated so that GPC will be notified that reset assertion is performed and GPC can continue with release of ISO signal.

Upon deassertion of gpc_reset_vpu_req (gpc_reset_vpu_req=’1’) and the completion of the above procedure then perform the following steps:

1. Deassert vpu reset (vpu_rst_b=’1’)
2. Count atleast 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open vpu clocks (en_vpu_clk=’1’).
4. Count atleast 16 ipg_clk clocks so that reset will propagate in vpu
5. Assert vpu_reset_done internally to SRC.

If the request came from software bit, i.e. sw_vpu_rst = ‘1’ then perform the following steps:

1. Close vpu clocks (en_vpu_clk=’0’)
2. Assert vpu reset (vpu_rst_b=’0’)
3. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
4. Open vpu clocks (en_vpu_clk=’1’)
5. Count at least 16 ipg_clk clocks so that reset will propagate in vpu
6. Close vpu clocks (en_vpu_clk=’0’)
7. Wait at least 8 ipg_clk cycles so that clocks to vpu will be closed.

SRC will continue to the deassertion steps immediately.

1. Deassert vpu reset (vpu_rst_b=’1’)
2. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open vpu clocks (en_vpu_clk=’1’).
4. Count at least 16 ipg_clk clocks so that reset will propagate in vpu
5. Generate interrupt (if its not masked) to notify that reset vpu passed reset and is ready to be used and clear sw_vpu_rst.

Figure 54-17 describes the flow of vpu reset pulse generator.

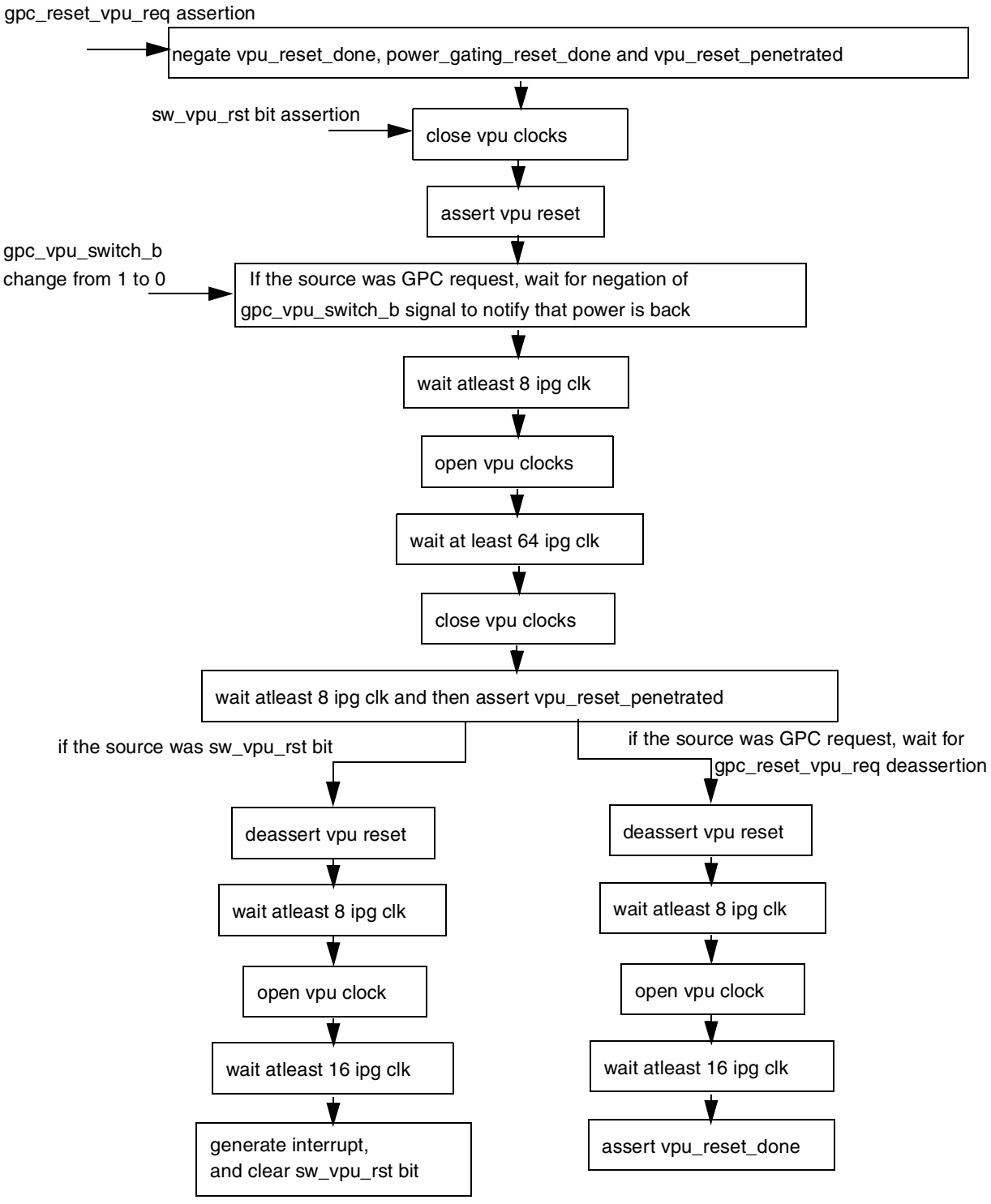


Figure 54-17. vpu Reset Pulse Generator

GPU reset pulse generator:

GPU reset pulse can be generated from sw_gpu_rst bit or gpc_reset_gpu_req.

If the request came from gpc, i.e. gpc_reset_gpu_req=’0’ then negate gpu_reset_done internally to SRC. This will cause negation of power_gating_reset_done signal to CCM. On this stage also deassert gpu_reset_penetrated.

1. Close gpu clocks (en_gpu_clk=’0’) (just to make sure they are closed)
2. Assert gpu reset (gpu_rst_b=’0’).
3. If the source was GPC request, wait for change from 1 to 0 on gpc_gpu_switch_b signal to notify that power is back.
4. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
5. Open gpu clocks.
6. Count at least 16 ipg_clk clocks so that reset will propagate in gpu (this stage will happen only after pll relock and clocks are ready prior to exiting from stop)
7. Close gpu clocks (en_gpu_clk=’0’)
8. Wait at least 8 ipg_clk cycles so that clocks to gpu will be closed, and then assert gpu_reset_penetrated so that GPC will be notified that reset assertion is performed and GPC can continue with release of ISO signal.

Upon de assertion of gpc_reset_gpu_req (gpc_reset_gpu_req=’1’) and finish of the above procedure then:

1. Deassert gpu reset (gpu_rst_b=’1’)
2. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open gpu clocks (en_gpu_clk=’1’).
4. Count at least 16 ipg_clk clocks so that reset will propagate in gpu
5. Assert gpu_reset_done internally to SRC.

If the request came from software bit, i.e. sw_gpu_rst = ‘1’ then perform the following steps:

1. Close gpu clocks (en_gpu_clk=’0’)
2. Assert gpu reset (gpu_rst_b=’0’)
3. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
4. Open gpu clocks (en_gpu_clk=’1’)
5. Count at least 16 ipg_clk clocks so that reset will propagate in gpu
6. Close gpu clocks (en_gpu_clk=’0’)
7. Wait a tleast 8 ipg_clk cycles so that clocks to gpu will be closed.

SRC will continue to the deassertion steps imediately.

1. Deassert gpu reset (gpu_rst_b=’1’)
2. Count atleast 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open gpu clocks (en_gpu_clk=’1’).
4. Count atleast 16 ipg_clk clocks so that reset will propagate in gpu

5. Generate interrupt (if its not masked) to notify that reset gpu passed reset and is ready to be used and clear sw_gpu_rst.

Figure 54-18d escribes the flow of gpu reset pulse generator:

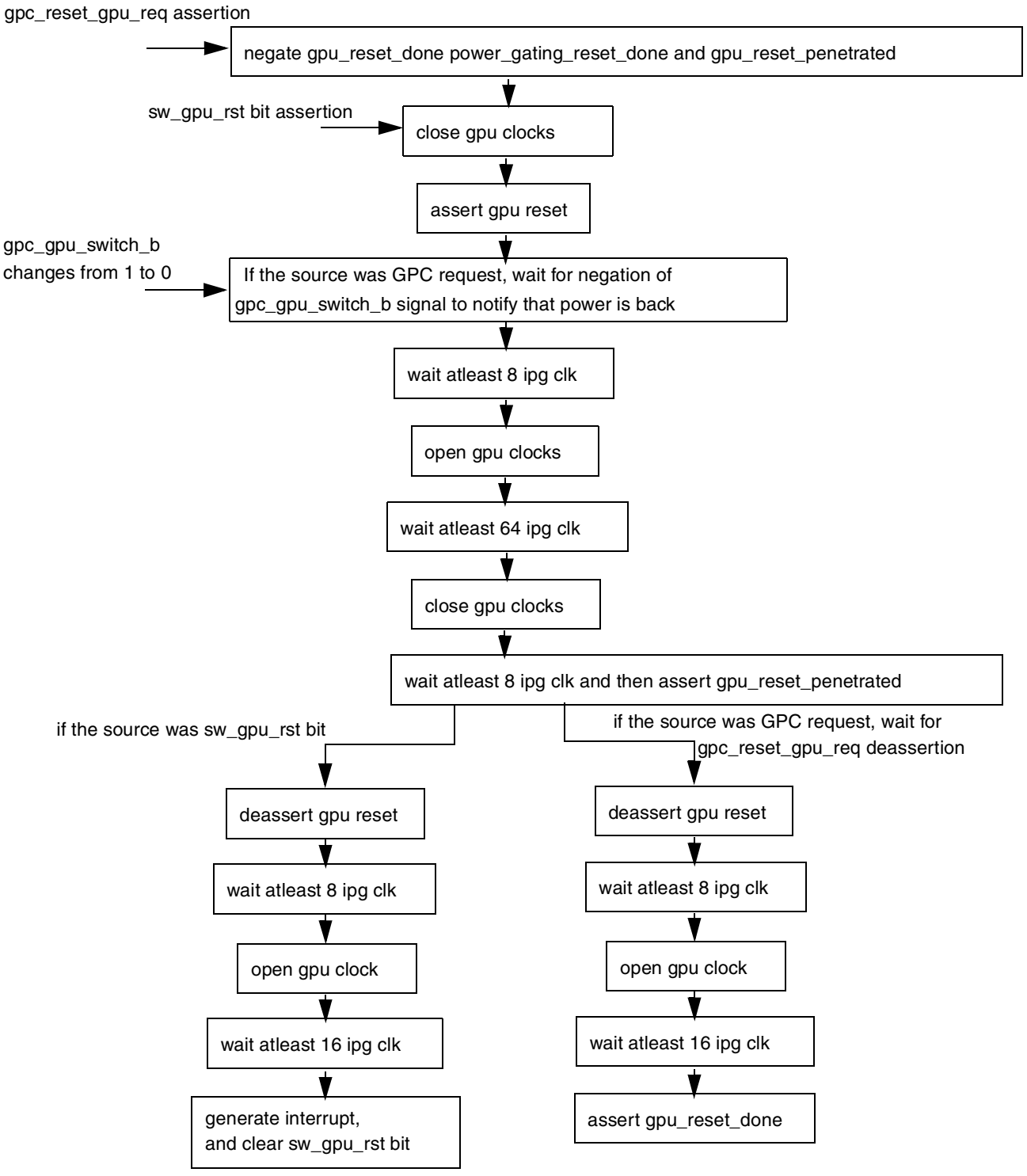


Figure 54-18. gpu Reset Pulse Generator Flow

Open_VG reset pulse generator:

Open_VG reset pulse can be generated from sw_open_vg_rst bit or gpc_reset_open_vg_req.

If the request came from gpc, i.e. gpc_reset_open_vg_req=’0’ then negate open_vg_reset_done internally to SRC. This will cause negation of power_gating_reset_done signal to CCM. On this stage also deassert open_vg_reset_penetrated.

1. Close open_vg clocks (en_open_vg_clk=’0’) (just to make sure they are closed)
2. Assert open_vg reset (open_vg_rst_b=’0’).
3. If the source was GPC request, wait for change from 1 to 0 on gpc_open_vg_switch_b signal to notify that power is back.
4. Count atleast 8 ipg_clk clocks so that reset will propagate the reset tree
5. Open open_vg clocks.
6. Count at least 16 ipg_clk clocks so that reset will propagate in open_vg (this stage will happen only after pll relock and clocks are ready prior to exiting from stop)
7. Close open_vg clocks (en_open_vg_clk=’0’)
8. Wait at least 8 ipg_clk cycles so that clocks to open_vg will be closed, and then assert open_vg_reset_penetrated so that GPC will be notified that reset assertion is performed and GPC can continue with release of ISO signal.

Upon de assertion of gpc_reset_open_vg_req (gpc_reset_open_vg_req=’1’) and finish of the above procedure then:

1. Deassert open_vg reset (open_vg_rst_b=’1’)
2. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open open_vg clocks (en_open_vg_clk=’1’).
4. Count at least 16 ipg_clk clocks so that reset will propagate in open_vg.
5. Assert open_vg_reset_done internally to SRC.

If the request came from software bit, i.e. sw_open_vg_rst = ‘1’ then:

1. Close open_vg clocks (en_open_vg_clk=’0’)
2. Assert open_vg reset (open_vg_rst_b=’0’)
3. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
4. Open open_vg clocks (en_open_vg_clk=’1’)
5. Count at least 16 ipg_clk clocks so that reset will propagate in open_vg
6. Close open_vg clocks (en_open_vg_clk=’0’)
7. Wait at least 8 ipg_clk cycles so that clocks to open_vg will be closed.

SRC will continue to the deassertion steps imediately.

1. Deassert open_vg reset (open_vg_rst_b=’1’)
2. Count at least 8 ipg_clk clocks so that reset will propagate the reset tree
3. Open open_vg clocks (en_open_vg_clk=’1’).

4. Count at least 16 ipg_clk clocks so that reset will propagate in open_vg.
5. Generate interrupt (if its not masked) to notify that reset open_vg passed reset and is ready to be used and clear sw_open_vg_rst.

Figure 54-19 describes the flow of open_vg reset pulse generator:

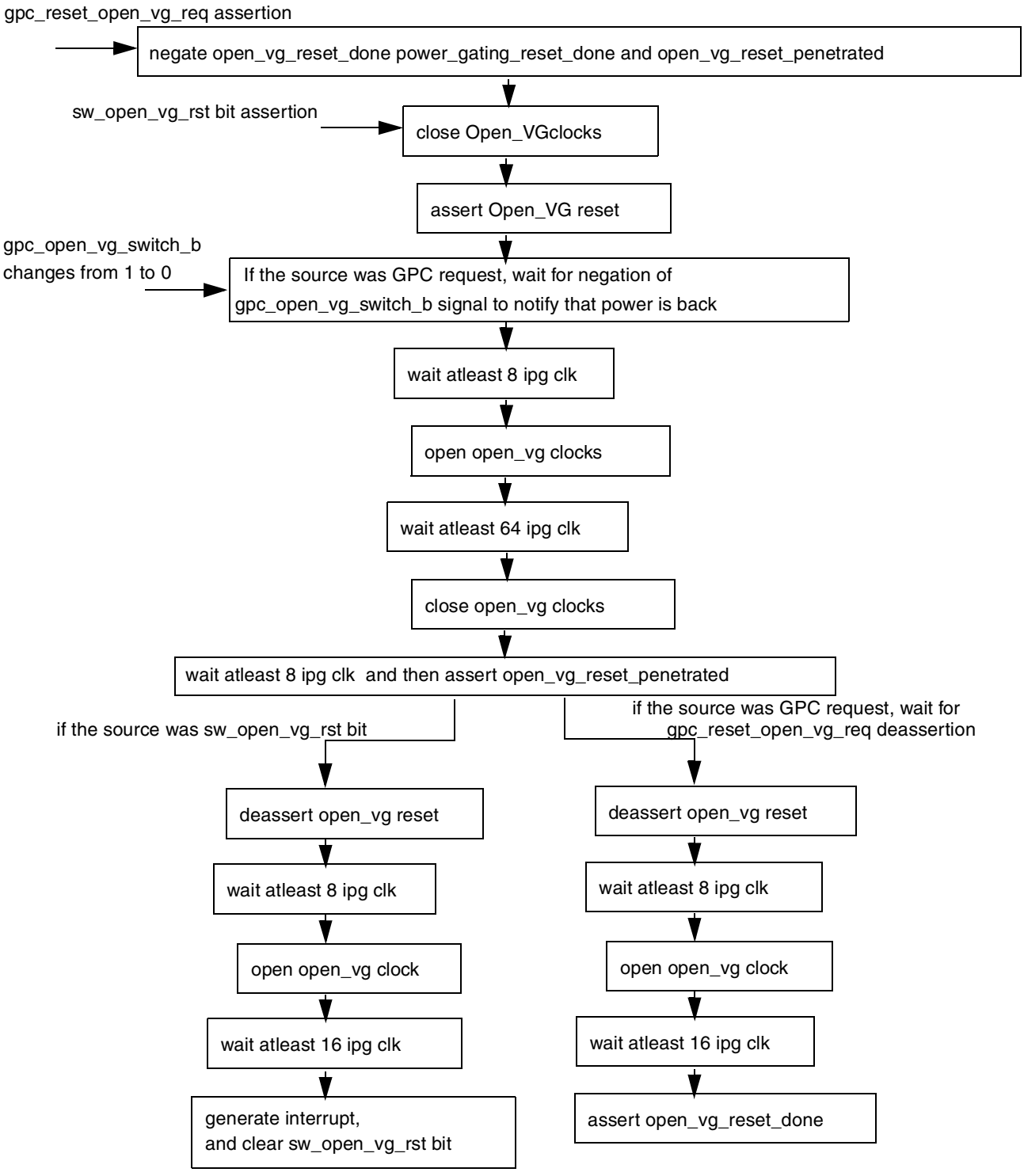


Figure 54-19. open_vg Reset Pulse Generator Flow

Upon assertion of ipu_reset_done, vpu_reset_done, open_vg_reset_done and gpu_reset_done, SRC will generate assertion of power_gating_reset_done to notify ccm that all needed resets has finished and ccm can continue with waking the system from stop mode, i.e enabling arm and system's clocks.

54.3.1.2.5 SRTC_RST_B generation

SRTC_RST_B will be generated during POR sequence together with test_logic_rst_b.

On the srtc_rst_b path there will be a mux that enables a generation of reset on srtc_rst_b pad from TCU , in case of test_mode. The enable of this mux is test_mode input. When the test_mode input equals '0' , the generated srtc_rst_b will be the functional srtc reset (generated during POR scenario). When the test_mode input equals '1', the generated srtc_rst_b will be connected to the tcu_reset input. In this case, tcu will have complete controlability of srtc_rst_b , and will be able to generate a reset for srtc.

Figure 54-20 describes this mux.

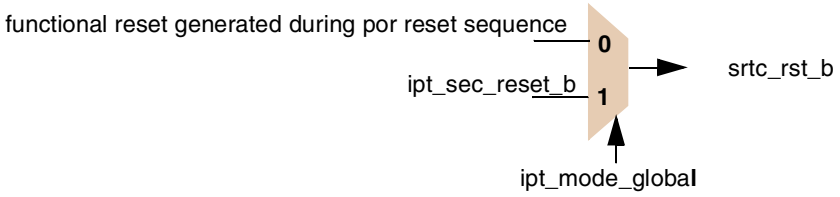


Figure 54-20. srtc_rst_b Generation

54.3.2 SJC_POR_RST_B Generation

SJC will be reseted by sjc_por_rst_b output of SRC. This output will be toggled only on POR sequence together with test_logic_rst_b. It will not be toggled on IEEE reset (jtag_rst_b).

54.3.3 SRC_MEGAMIX_ISO Generation

In normal functional mode (after system came out of reset), the GPC is controling the ISO signal for the isolation cells that wrap the Megamix. During reset sequence, SRC will drive the ISO signal for those isolation cells. SRC will drive those isolation cells by generation of src_megamix_iso output signal. This signal will equal to '0' during reset sequence, until deassertion of "reset_all_modules" (untill reset_all_modules='1'). After the deassertion of "reset_all_modules", the value of the ISO signal generated from GPC (gpc_megamix_iso) will be assigned to the output src_megamix_iso, and delivered to the isolation cells wraping the megamix.

54.3.4 Parallel Reset Requests

This chapter describes the behavior when paralel reset requests are received. In the parallel reset requests case SRC follows the following rules:

- The order of strength of resets is POR = strongest, cold = medium, and warm = weakest.
- If a stronger reset is asserted during a weaker reset's sequence, the stronger reset takes over and begins its reset process. The following cases fall into this category:
  - POR reset request in the middle of cold or warm reset process—the cold or warm reset process will be stopped and the POR sequence will start.
  - COLD reset request in the middle of warm reset process—the warm reset process will be stoped and the cold sequence will start.
- If a weaker reset is asserted during stronger reset sequence, the stronger reset sequence continues without interference. If at the end of the stronger reset process, the weaker request is still asserted then the weaker sequence will commence. The following cases fall into this category:
  - COLD or WARM reset requests in the middle of POR reset process—the POR process continues without interference.
  - WARM reset request in the middle of COLD reset process—the COLD process continues without interference.
- If a similar reset request is asserted during the process of reset handling, then the process of reset handling will start over (with the same process). The following cases fall into this category:
  - POR reset request in the middle of POR reset process—the POR process starts over.
  - COLD reset request in the middle of COLD reset process—the COLD process starts over.

There is one exception to this category: a WARM reset request in the middle of WARM reset process. In this case, the new WARM reset process cannot restart because EMI is reset. There cannot be the required handshake with EMI when EMI is reset. For this case, the first WARM reset will continue. Only if the second WARM reset is still asserted after the first one has finished, does the WARM sequence begin again.

54.3.5 DFT Mux on Reset Outputs

All functional reset output described above are muxed before exiting SRC. The mux is controlled by ipt_mode_global signal. The mux chooses between the functional reset and the DFT reset, as shown in Figure 54-21.

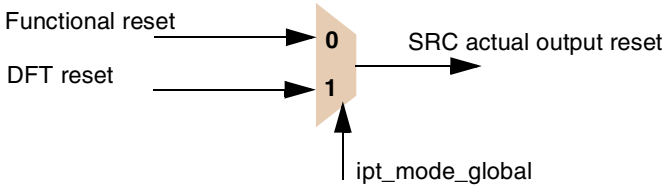


Figure 54-21. MUX Between Functional Reset and DFT Reset

Table 54-11 describes the DFT reset assigned to each of the reset outputs.

Table 54-11. DFT Reset Assigment

SRC Reset Output	DFT Reset Generation
arm_por_rst	~(ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b)
arm_soc_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
dftrst	~(ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b)
dftrst_vpu	~(ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b)
dftrst_ipu	~(ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b)
emi_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
ipu_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
rst_sms_n	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
rst_sms_n_vpu	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
rst_sms_n_ipu	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
system_early_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
system_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
test_logic_rst_b	ipp_reset_b
vpu_rst_b	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
wrst_n	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
wrst_n_vpu	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
wrst_n_open_vg	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
wrst_n_gpu	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
wrst_n_ipu	ipp_reset_b & ipp_user_reset_b & ipt_sec_reset_b
srtc_rst_b	ipt_sec_reset_b
src_por_b	ipp_reset_b

Note: src_por_b is an internal reset generated for SRC module only. This reset is asserted on por only (ipp_reset_b). It is equal to the functional test_logic_reset_b.

54.3.6 Boot Mode Control

54.3.6.1 BMOD Pin Latching

The exact boot sequence for the i.MX51 is controlled by the values of the BMOD pins on this device.

All the boot pins will be sampled at deassertion of system_early_rst_b.

The value of the BMOD pins will be latched after the IIM asserts the fuse read completion flag. After latching, the values of the BMOD pins are used to determine the booting options of the core as given in the SBMR register.

The information for DIR_BT_DIS is programed in the fuse box.

The Boot mode information for MEM_TYPE, PG_SIZE, BUS_WIDTH, MEM_CTL can be provided to the SRC from either e-fuses or GPIO signals. The gpio_bt_sel e-fuse defines the source to be used to derive the boot information. When gpio_bt_sel is set, e-fuses are used. When cleared, GPIO signals are used.

When FSL test mode is chosen (01 for BMOD[1:0]) then the Fuses signals are used.

When Internal Boot From Fuses mode is chosen (10 for BMOD[1:0]) then e-fuses signals are used, without consideration on the gpio_bt_sel e-fuse value.

The boot information is provided in the SBMR register. Figure shows the selection of Boot mode information.

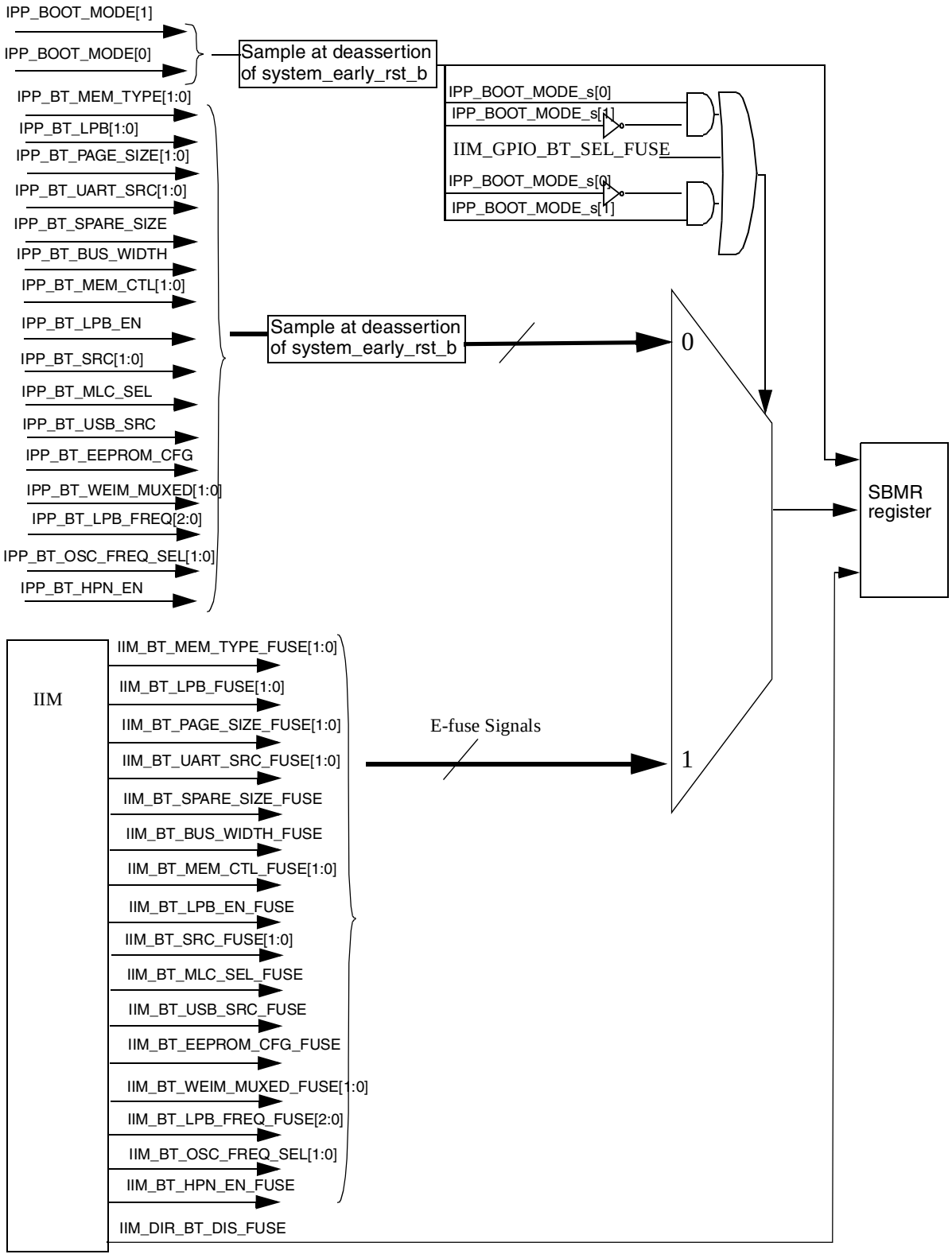


Figure 54-22. Boot Mode Information

Note: IPP signals that need to be latched by SRC will be set to the corresponding mode based on system_rst_b signal. When system_rst_b signal equals ‘0’, the IOMUX will shift to the needed mode for transfering boot values on the IO pins. When system_rst_b signal equals ‘1’, the IOMUX will shift to the default mode.

54.3.6.2 Output Boot Signals

The output boot signals are as follows:

- src_int_boot—internal boot indicator: Boot mode = External + WEIM mem type, are counted as external mode for ARM.
  - assign src_int_boot = iim_dir_bt_dis_fuse || ~(ipp_boot_mode[1:0]=01 && test_mode[2:0]=3’b111 && (BT_MEM_CTL[1:0]==2'b00))
- src_weim_boot_cfg[2] - Multiplexed address mode: To be driven based on SBMR bit [16] value. (0-not muxed, 1-Multiplexed mode). Connect to EMI pin src_weim_boot_cfg[2].
  - src_weim_boot_cfg[1]= SBMR[16]&SBMR[17] | ~SBMR[16]&~SBMR[17] ;
  - src_weim_boot_cfg[0]= SBMR[16] | SBMR[17] ;
- src_boot_add[31:0] - boot address - output to ARM Platform
  - IF src_int_boot=0 then src_boot_add[31:0]=32’h B000_0000
  - Otherwise src_boot_add[31:0]=32’h 0000_0000
- ipp_do_boot_mode_inv[1:0] - inversion of IPP_BOOT_MODE[1:0]. Once the boot signals (ipp_boot_mode[1:0]) are sampled in SRC, SRC will generate the inversion of ipp_boot_mode[1:0] on those pins. The IO ring will use system_rst_b to generate the ipp_do_boot_mode_inv[1:0] on the boot pads:

When system_rst_b=0, the boot pads will be configured as inputs, allowing boot info to propagate to SRC’s inputs ipp_boot_mode[1:0].

When system_rst_b=1 , the boot pads will be configured as output, allowing ipp_do_boot_mode_inv[1:0] to be generated to the pads, i.e. it will be noticed on the pads that the value has flipped.

The board will catch this flip and in this way will be notified that the system sequence has changed, and it is allowed to use the boot pads for a different purpose.

Chapter 55
Secure Real Time Clock (SRTC)

55.1 Overview

The SRTC helps to comply with issues arising out of different application requiring secure and certifiable time for example Digital Rights Management (DRM) schemes.

DRM are used to offer access to digital content, by a consumer, while still offering protection to the owner. The consumer can only use the digital content in authorized ways. There are typically rights rules associated with the content defining how it can be used, or for how long. One of the right rules associated with the content is a time based usage. For example, a sample of music could be allowed to be played for only one week. If the consumer were able to modify the time source to act as if it were last week, then the DRM hardware would allow the content to be used for an extra week.

In order to enforce those rules, trusted hardware is required. Otherwise, the rights rules might be bypassed, and the content used in ways not desired by the owner. To be able to enforce a time restriction on the use of the content, trusted time source is required. This trusted time source must provides features that allow the system software designer to ensure that the time kept by the device is certifiable. SRTC provides a certifiable time to user and can raise alarm if tampering with counters is detected.

The SRTC is comprised of two separated submodules:

- Lower power domain SRTC (SRTC LP)
- High power domain SRTC (SRTC HP)

SRTC LP block is in always powered up domain and is isolated from the rest of the logic by means of isolation cell. Isolations cells are library instantiated cells, which insure that the powered up logic does not get corrupted when the power goes down in the rest of the SoC. It is powered by the Coin Cell battery

SRTC HP is in the normal supply domain. It is powered by the main digital supply powering the SoC

55.1.1 Low Power SRTC (SRTC LP) Description

The SRTC LP is a real time clock with enhanced security capabilities. Its purpose is to provide an accurate time read at all time, regardless of the main system power state and without the need to use external on-board time source such as external RTC. The SRTC LP can wake up the system when preset time alarm is asserted.It is divided into following units:

- Counters and registers
  - — Non-rollover 47-bit time counter
  - — 32-bit alarm register

  - General purpose registers
  - Secured monotonic counter
- SRTC Monitor
  - SRTC state machine
  - SRTC power and clock detectors

See Figure 55-1 for the SRTC module partitioning.

55.1.2 High Power SRTC (SRTC HP) Description

The SRTC HP contains all the interfaces to the core (IPBus interface). It contains counter, alarm and has the capability of generating a number of user defined periodic interrupt. Access to SRTC LP registers can only be done through the SRTC HP and when SRTC HP is powered up.

SRTC HP is partitioned into following units.

- Address Decode
- Counters and registers
  - Non-secure 47-bit time counter
  - Programmable non-secure 47-bit alarm with interrupt
  - Periodic alarm with interrupt

See Figure 55-1 for the SRTC module partitioning.

The non secure time counter can be synchronized with the secured time counter by writing to a SRTC HP bit (time_sync-Time Synchronization). When written, the content of the secure counter is automatically copied to the non-secure counter

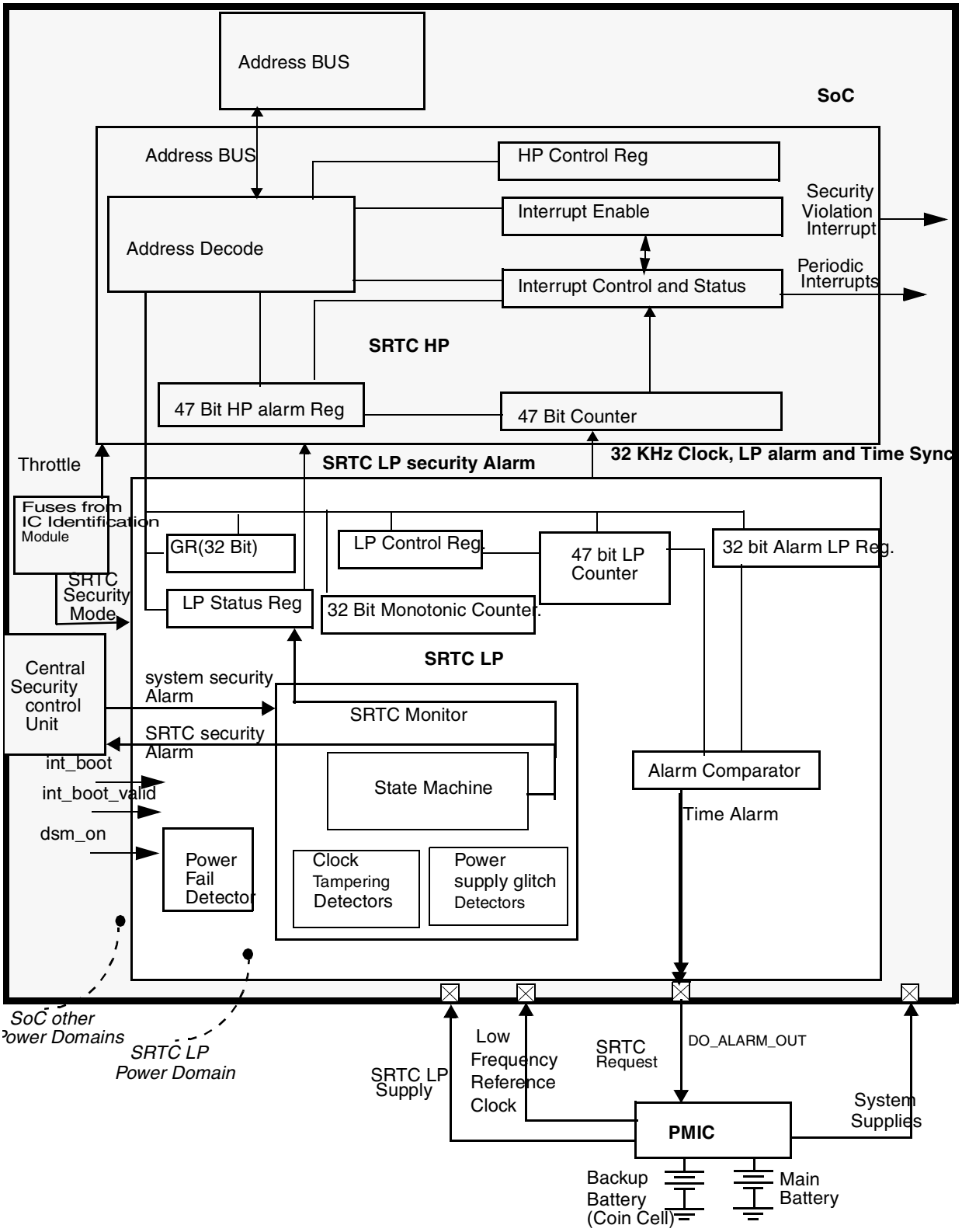


Figure 55-1. Block Diagram

55.1.3 Features

The SRTC modules includes the following features:

- Secure 47-bit time counter running on 32.768KHz clock
- Non-secure 47-bit time counter running on 32.768KHz clock
- User mode protection and/or ARM's TrustZone mode protection-
  SRTC secured registers cannot be configured by non-secured SW . A dedicated control bit can allow non-secured SW to update the secured registers. This bit can be set only by secured SW
  In any case, SRTC secured registers cannot be configured when SRTC is locked.
- Re-programming protection:
  - SRTC time cannot be altered after SRTC is locked (when SRTC set to the appropriate security mode)
  - SRTC cannot be disabled after SRTC is locked (when SRTC set to the appropriate security mode)
- Power loss protection:
  - Separated power supply
  - Accurate continuous time is kept during power down system states and main power source losses
  - In the event of a total power loss (main power source and backup power source), SRTC time read is invalidated
  - In the event of unstable voltage power source, SRTC time read is invalidated
  - In the event of low voltage power source that disrupts normal time counting, SRTC time read is invalidated
- Clock source protection:
  - Directly connects to a low frequency crystal (32.768 kHz) or to a PMIC 32.768 kHz clock source
  - In the event that SRTC frequency drops, SRTC time read is invalidated
- SRTC time can be invalidated by a system security alarm
- SRTC can assert a system security alarm and fast interrupt request to alert the processor that time read was invalidated
- Programmable secure 32-bit alarm with interrupt (alarm of the secure- LP section). In case of system power up mode, this alarm generates interrupt to alert the processor. In case of system power down mode, this alarm alert the power management IC (PMIC) via an external pin to instruct the PMIC to power up the system. Note that this feature does not require any PMIC special support. This signal can be connected to an on-board power on button circuit.
- Programmable non-secure 47-bit alarm with interrupt (alarm of the non-secure, HP, section)
- Periodic non-secure alarm with interrupt
- Secured monotonic counter
- General purpose "always powered" registers

- Ultra low power consumption. SRTC LP power consumption during system power down state is less than 3 µA.(TYP)
- Separate calibration logic for 47 bit HP and LP counters

55.1.4 Modes of Operations

The SRTC has three functional modes, defined by IC Identification Module dedicated fuses, and can be found in one of four states.

The SRTC modes are as follows.

- Mode #1—High Security
- Mode #2—Medium Security
- Mode #3—Low Security

SRTC states are defined as follows.

- “Initialization”—SRTC is powered up for the first time or after system POR.
- “Non valid”—SRTC time is not set, time read is not authenticated
- “Valid”—Valid time was set, time read is authentic

NOTE

For detailed information about the SRTC, contact your Freescale representive.

Chapter 56
Synchronous Serial Interface (SSI)

This chapter describes the Synchronous Serial Interface (SSI). It discusses the architecture, the programming model, the operating modes, and initialization of SSI.

56.1 Overview

The SSI is a full-duplex, serial port that allows the chip to communicate with a variety of serial devices. These serial devices can be standard CODer-DECoder (CODECs), Digital Signal Processors (DSPs), microprocessors, peripherals, and popular industry audio CODECs that implement the inter-IC sound bus standard (I^2S) and Intel AC97 standard.

SSI is typically used to transfer samples in a periodic manner. The SSI consists of independent transmitter and receiver sections with independent clock generation and frame synchronization.

See Figure 56-1 for illustration of the block diagram for the SSI. It consists of control registers to set up the port, status register, separate transmit and receive circuits with FIFO registers, and separate serial clock and frame sync generation for the transmit and receive sections. The second set of Tx and Rx FIFOs, replicates the logic used for the first set of FIFOs.

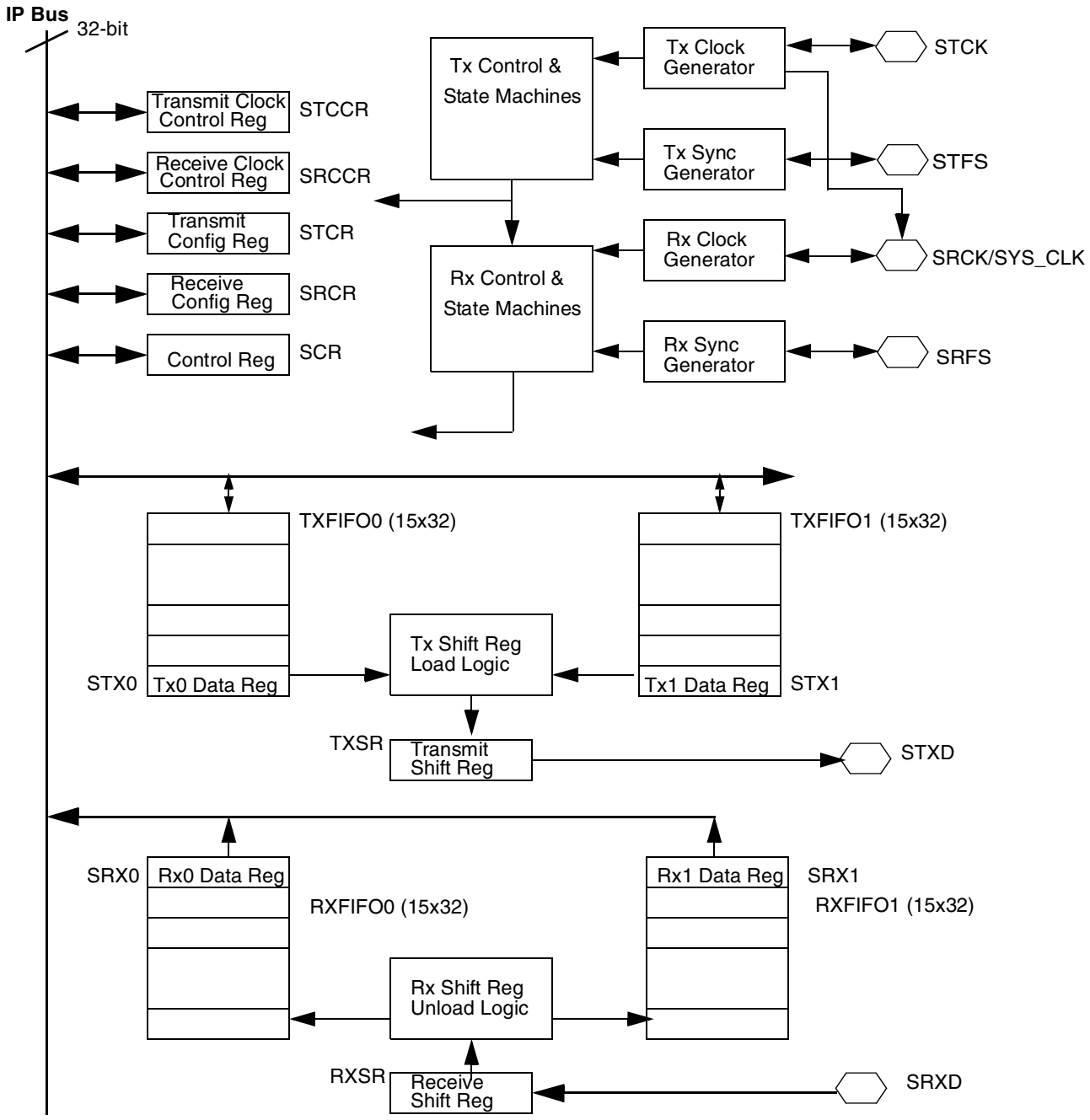


Figure 56-1. SSI Block Diagram

56.1.1 Features

The SSI includes the following features:

- Independent (asynchronous) or shared (synchronous) transmit and receive sections with separate or shared internal/external clocks and frame syncs, operating in Master or Slave mode.
- Normal mode operation using frame sync

- Network mode operation allowing multiple devices to share the port with as many as thirty-two time slots
- Gated Clock mode operation requiring no frame sync
- Two sets of Transmit and Receive FIFOs. Each of the four FIFOs is 15 × 32 bits. The two sets of Tx/Rx FIFOs can be used in Network mode to provide two independent channels for transmission and reception
- Programmable data interface modes such like I2S, LSB, MSB aligned
- Programmable word length (8, 10, 12, 16, 18, 20, 22 or 24 bits)
- Program options for frame sync and clock generation
- Programmable I2S modes (Master, Slave or Normal). Oversampling clock, ccm_ssi_clk available as output from SRCK in I2S Master mode
- AC97 support
- Completely separate clock and frame sync selections for the receive and transmit sections. In AC97 standard, the clock is taken from an external source and frame sync is generated internally.
- External ccm_ssi_clk input for use in I2S Master mode. Programmable oversampling clock (ccm_ssi_clk) of the sampling frequency available as output in master mode at SRCK, when operated in sync mode.
- Programmable internal clock divider
- Time Slot Mask Registers for reduced CPU overhead (for Tx and Rx both)
- SSI power-down feature
- Programmable wait states for CPU accesses
- IP Interface for register accesses, compliant to SRS 3.0.2 standard

56.1.2 Modes of Operation

SSI has the following basic operating modes.

- Normal mode
  - — Asynchronous protocol
  - — Synchronous protocol
- Network mode
  - — Asynchronous protocol
  - — Synchronous protocol
- Gated Clock mode
  - — Synchronous protocol only

These modes can be programmed by several bits in the SSI control registers. See Table 56-1 for the list of SSI operating modes and some of the typical applications in which they can be used:

Table 56-1. SSI Operating Modes

TX, RX Sections	Serial Clock	Mode	Typical Application
Asynchronous	Continuous	Normal	Multiple synchronous CODECs
Asynchronous	Continuous	Network	TDM CODEC or DSP networks
Synchronous	Continuous	Normal	Multiple synchronous CODECs
Synchronous	Continuous	Network	TDM CODEC or DSP network
Synchronous	Gated	Normal	SPI-type devices; DSP to MCU

The transmit and receive sections of the SSI can be synchronous or asynchronous. In Synchronous mode, the transmitter and the receiver use a common clock and frame synchronization signal. Masking of slots for Transmit and Receive section can differ in synchronous mode. Also the shifting of data in independent and for receive section depends on RXBIT0 and RSHFD bits in SRCR register. In Asynchronous mode, the transmitter and receiver each has its own clock and frame synchronization signals.

Normal or Network mode can be selected. In Normal mode, the SSI functions with one data word of I/O per frame. In Network mode, any number from two to thirty-two data words of I/O per frame can be used. Network mode is typically used in star time division multiplex networks with other processors or CODECs, allowing interface to time division multiplexed networks without additional logic. Gated clock mode option can be selected in Normal synchronous mode only. During Gated-mode the clock is not-continuous and runs only during data-transmission. These distinctions result in the basic operating modes that allow the SSI to communicate with a wide variety of devices.

The SSI supports both Normal and Network modes, and these can be selected independently of whether the transmitter and receiver are synchronous or asynchronous. Typically these protocols are used in a periodic manner, where data is transferred at regular intervals, such as at the sampling rate of an external CODEC. Both modes use the concept of a frame. The beginning of the frame is marked with a frame sync when programmed with continuous clock. The frame sync occurs at a periodic interval. The length of the frame is determined by the DC[4:0] bits in either the SRCCR or STCCR register, depending on whether data is being transmitted or received. The number of words transferred per frame depends on the mode of the SSI.

In Normal mode, one data word is transferred per frame. In Network mode, the frame is divided into anywhere between two and thirty-two time slots, where in each time slot one data word can optionally be transferred.

Apart from the above basic modes of operation, SSI supports the following modes which require some specific programming.

- I2S mode
- AC97 mode
  - AC97 Fixed mode
  - AC97 Variable mode

In (non-I2S) slave modes (external frame sync), the programmed word length setting of the SSI should be equal to the word length setting of the master. In I2S slave mode, the programmed word length setting of the SSI can be lesser than or equal to the word length setting of the I2S master (external CODEC).

In slave modes, the programmed frame length setting (DC bits) of the SSI can be lesser than or equal to the frame length setting of the master (external CODEC).

The following sections provide detailed descriptions of the above modes.

56.1.2.1 Normal Mode

Normal mode is the simplest mode of the SSI. It is used to transfer data in one time slot per frame. A time slot is a unit of data and the WL[3:0] bits define the number of bits in a time slot. In Continuous Clock mode, a frame sync occurs at the beginning of each frame. The length of the frame is determined by the following factors:

- The period of the Serial Bit Clock (DIV2, PSR, PM[7:0] bits for internal clock or the frequency of the external clock on the STCK port)
- The number of bits per time slot (WL[3:0] bits)
- The number of time slots per frame (DC[4:0] bits)

If Normal mode is configured with more than one time slot per frame, data is transferred only in the first time slot. No data is transferred in subsequent time slots. In Normal mode, DC[4:0] values corresponding to more than a single time slot in a frame, only results in lengthening of the frame.

56.1.2.1.1 Normal Mode Transmit

The conditions for data transmission from the SSI in Normal mode are:

- SSI Enabled (SSIEN = 1)
- Write data to Transmit Data Register (STX)
- Transmitter Enabled (TE = 1)
- Frame sync active (for continuous clock case)
- Bit clock begins

When the above conditions occur in Normal mode, the next data word is transferred into the Transmit Shift Register (TXSR) from the Transmit Data Register 0 (STX0), or from the Transmit FIFO 0 Register, if transmit FIFO 0 is enabled. The new data word is transmitted on arrival of frame-sync preceded by clocks in continuous clock mode. In gated external mode, data word in transmitted on arrival of frame-sync. In gated-internal mode, data word is transmitted whenever data is available in Tx-FIFO.

If Transmit FIFO 0 is not enabled and the transmit data register empty enable (TDE0_EN) and transmit interrupt enable (TIE) bits are set, transmit interrupt occurs when the word in SSI_STX0 is shifted to Transmit Shift (TXSR) register for shifting.

If Transmit FIFO 0 is enabled and the transmit fifo full enable (TFF0_EN) and transmit interrupt enable (TIE) bits are set, transmit interrupt occurs when the number of empty slots in Transmit Fifo 0 exceed or are equal to the selected threshold value i.e. Transmit Fifo 0 Watermark (TFWM0) value. If transmit FIFO

0 is enabled and filled with data, 15 data words can be transferred before the core must write new data to the STX0 register.

The STXD port is disabled except during the data transmission period. For a continuous clock, the optional frame sync output and clock outputs are not disabled, even if both receiver and transmitter are disabled.

56.1.2.1.2 Normal Mode Receive

The conditions for data reception from the SSI are:

- SSI enabled (SSIEN = 1)
- Receiver enabled (RE = 1)
- Frame sync active (for continuous clock case)
- Bit clock begins

With the above conditions in Normal mode with a continuous clock, each time the frame sync signal is generated (or detected) a data word is clocked in. With the above conditions and a gated clock, each time the clock begins, a data word is clocked in.

If Receive FIFO 0 is not enabled and Receive Interrupt enable (RIE) and Received Data 0 Ready enable (RDR0_EN) bits are set, receive interrupt occurs when received data word is transferred from the Receive Shift Register (RXSR) to the Receive Data Register 0 (SRX0) thus setting the Receive Data Ready 0 (RDR0) flag.

If Receive FIFO 0 is enabled, and Receive Interrupt enable (RIE) and Received Fifo 0 full enable (RDR0_EN) bits are set, receive interrupt occurs when the received data word is transferred to the Receive FIFO 0 and Receive FIFO 0 reaches the selected threshold and results in Receive FIFO Full 0 (RFF0) flag to get set.

The core program has to read the data from the Receive Data Register 0 (SRX0) (in case Receive FIFO0 is disabled) before a new data word is transferred from the Receive Shift Register (RXSR), otherwise the Receive Overrun Error 0 (ROE0) bit is set. If receive FIFO 0 is enabled, the Receive Overrun Error 0 (ROE0) bit is set when the Receive FIFO 0 data level reaches the selected threshold and a new data word is ready to be transferred to the Receive FIFO 0.

See Figure 56-2 for illustration of transmitter and receiver timing for an 8-bit word in the first time slot in Normal mode, continuous clock with a late word length frame sync. The Tx Data register is loaded with the data to be transmitted. On arrival of the clock, this data is transferred to the Transmit Shift Register which gets transmitted on arrival of the frame-sync on the STXD output. Simultaneously, the Receive Shift

Register shifts in the received data available on the SRXD input and at the end of the time slot, this data is transferred to the Rx Data Register.

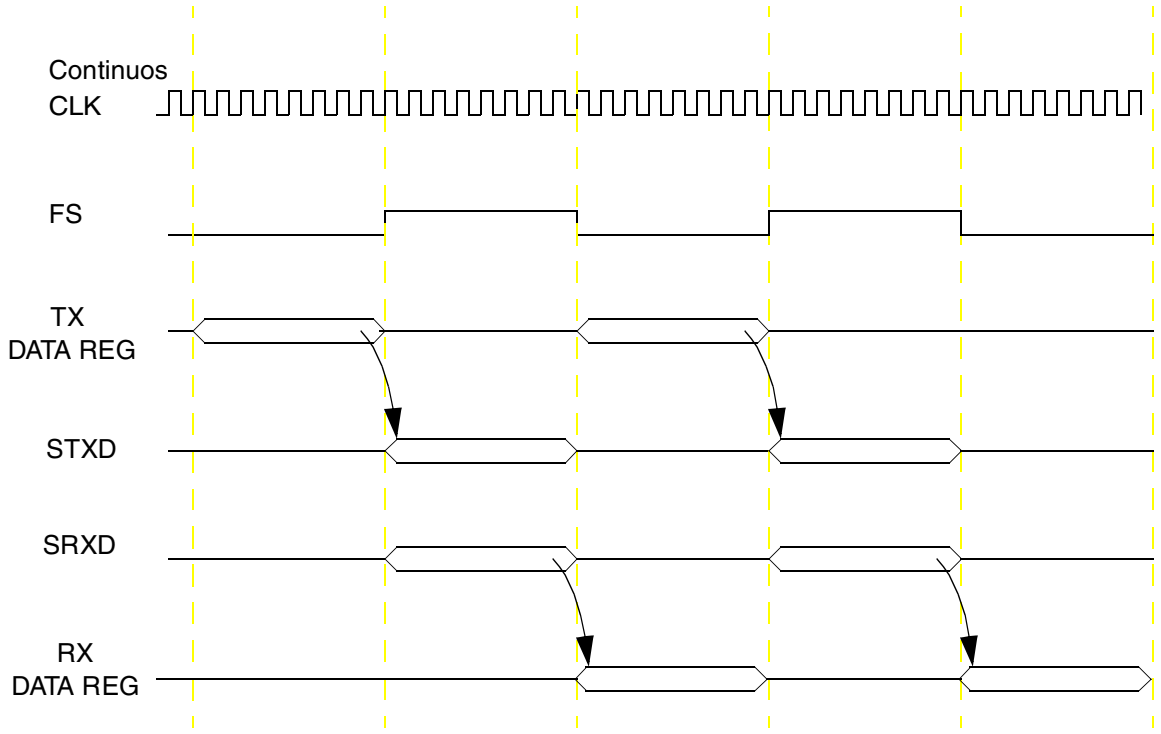


Figure 56-2. Normal Mode Timing - Continuous Clock

Figure 56-3 shows a similar case for internal (SSI generates clock) gated clock mode, and Figure 56-4 shows a case for external (SSI receives clock) gated clock mode.

NOTE

A pull-down resistor is required in the gated clock case because the clock port is disabled between transmissions.

The Tx Data register is loaded with the data to be transmitted. On arrival of the clock, this data is transferred to the transmit shift register which gets transmitted on arrival of the frame-sync on the STXD output. Simultaneously, the receive shift register shifts in the received data available on the SRXD input and at the end of the time slot, this data is transferred to the Rx Data Register. In case of Internal Gated clock mode, the Tx Data line and clock output port are put in the high-impedance state at the end of transmission of the last bit (at the completion of the complete clock cycle), whereas in External Gated clock mode, the Tx Data line is tri-stated at the last inactive edge of the incoming bit clock (during the last bit in a data word).

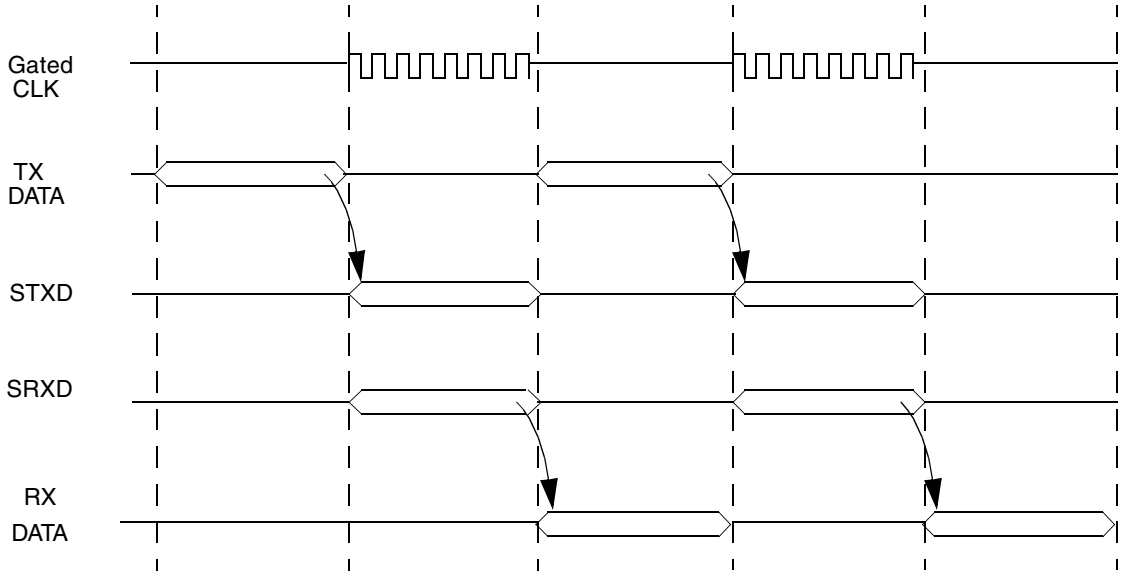


Figure 56-3. Normal Mode Timing—Internal Gated Clock

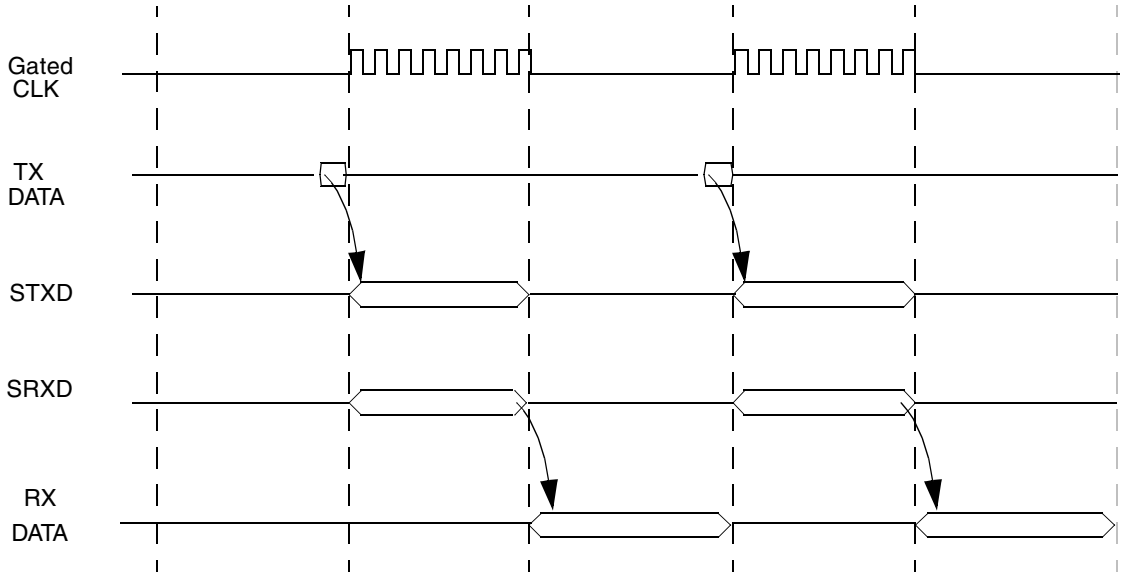


Figure 56-4. Normal Mode Timing—External Gated Clock

56.1.2.2 Network Mode

Network mode is used for creating a Time Division Multiplexed (TDM) network, such as a TDM CODEC network or a network of DSPs. In Continuous Clock mode, a frame sync occurs at the beginning of each frame. In this mode, the frame is divided into more than one time slot. During each time slot, one data word can be transferred. Each time slot is then assigned to an appropriate CODEC or DSP on the network. The DSP can be a master device that controls its own private network, or a slave device that is connected to an existing TDM network and occupies a few time slots.

The frame sync signal indicates the beginning of a new data frame. Each data frame is divided into time slots and transmission and/or reception of one data word can occur in each time slot (rather than in just the frame sync time slot as in Normal mode). The frame rate dividers, controlled by the DC[4:0] bits, select two to thirty-two time slots per frame. The length of the frame is determined by the following factors:

- The period of the serial bit clock (PSR, PM[7:0] bits for internal clock, or the frequency of the external clock on the STCK port)
- The number of bits per sample (WL[3:0] bits)
- The number of time slots per frame (DC[4:0] bits)

In Network mode, data can be transmitted in any time slot. The distinction of the Network mode is that each time slot is identified with respect to the frame sync (data word time). This time slot identification allows the option of transmitting data during the time slot by writing to the STX registers or ignoring the time slot as determined by STMSK register bits. The receiver is treated in the same manner and received data is only transferred to the receive data register/fifo if the corresponding time slot is enabled (through SRMSK).

By utilizing the STMSK and SRMSK registers, software only has to service the SSI during valid time slots. This eliminates any overhead associated with unused time slots. Refer to Section 56.3.3.20, SSI Transmit Time Slot Mask Register (STMSK)" and Section 56.3.3.21, SSI Receive Time Slot Mask Register (SRMSK)" for more information on STMSK and SRMSK.

In the Two-Channel mode of operation, the second set of Transmit and Receive FIFOs and Data Registers are used to create two separate channels. These channels are completely independent, with a their own set of Core interrupts and DMA requests, which are identical to the ones available for the default channel. In this mode, data is transmitted/received in enabled time slots alternately from/to FIFO 0 and FIFO 1, starting from FIFO 0. The first data word is taken from FIFO 0 and transmitted in the first enabled time slot and subsequently, data is loaded from FIFO 1 and FIFO 0 alternately and transmitted. Similarly, the first received data is sent to FIFO 0 and subsequent data is sent to FIFO 1 and FIFO 0 alternately. Time slots can be selected through the Transmit and Receive Time Slot Mask registers (STMSK and SRMSK). For using this mode of operation, the TCH_EN bit (SCR[8]) needs to be set.

56.1.2.2.1 Network Mode Transmit

The transmit portion of SSI is enabled when the SSIEN and the TE bits in the SCR are both set. However, for continuous clock, when the TE bit is set, the transmitter is enabled only after detection of a new frame sync (transmission starts from the next frame boundary).

Normal start-up sequence for transmission is to perform the following:

1. Enable Network Mode.
2. Enable SSI
3. Write the data to be transmitted to the STX register. This clears the TDE flag
4. Set the TE bit to enable the transmitter on the next frame boundary (for continuous clock case).
5. Enable transmit interrupts.

(Alternatively, the programmer may decide not to transmit in a time slot by configuring the STMSK.) TDE flag is cleared as data is shifted from STX register to TXSR, but the STXD port remains disabled during

the time slots. When the next frame sync is detected or generated (continuous clock), the data word in TXSR and is shifted out (transmitted). When the STX register is empty, the TDE bit is set, which causes a transmitter interrupt (in case the FIFO is disabled) to be sent if the TIE bit is set. Software can poll the TDE bit or use interrupts to reload the STX register with new data for the next time slot. Failing to reload the STX register before the TXSR is finished shifting (empty) causes a transmitter underrun and the TUE error bit is set. In case the FIFO is enabled, the TFE flag is set in accordance with the watermark setting and this flag causes the transmitter interrupt to occur.

The operation of clearing the TE bit disables the transmitter after completion of transmission of the current frame. Setting the TE bit enables transmission from the next frame. During that time the STXD port is disabled. The TE bit should be cleared after the TDE bit is set to ensure that all pending data is transmitted.

To summarize, the Network mode transmitter generates interrupts every enabled time slot (when FIFO is disabled) and requires the core program to respond to each enabled time slot in one of the following ways:

- Write data in data register to enable transmission in the next time slot.
- Configure the time slot register to disable transmission in the next time slot (unless time slot is already masked by STMSK register bit).
- Do nothing—transmit underrun occurs at the beginning of the next time slot and the previous data is re-transmitted.

In the Two-Channel mode of operation, both the channels (Data Registers, FIFOs, Interrupts and DMA requests) operate in the same manner, as described above. The only difference in case of the second channel is that the Interrupts related to this channel are generated only in case this mode of operation is selected (TDE1 is low by default).

56.1.2.2.2 Network Mode Receive

The receiver portion of the SSI is enabled when both the SSIEN and the RE bits in the SCR are set. However, the receive enable only takes place during that time slot if RE is enabled before the second to last bit of the word. If the RE bit is cleared, the receiver is disabled at the end of the current frame. SSI is capable of finding the start of the next frame automatically. When the word is completely received, it is transferred to the SRX register, which sets the RDR bit (Receive Data Ready). Setting the RDR bit causes a receive interrupt to occur if the receiver interrupt is enabled (the RIE bit is set) and (Receive data ready enable) RDR_EN bit is set. The second data word (second time slot in the frame) begins shifting in immediately after the transfer of the first data word to the SRX register. The core program has to read the data from the Receive Data Register (which clears RDR) before the second data word is completely received (ready to transfer to RX data register) or a receive overrun error occurs (the ROE bit is set).

An interrupt can occur after the reception of each enabled data word or the programmer can poll the RDR flag. The core program response can be one of the following:

- Read RX and use the data.
- Read RX and ignore the data.
- Do nothing—the receiver overrun exception occurs at the end of the current time slot.

NOTE

For a continuous clock, the optional frame sync output and clock output signals are not affected, even if the transmitter or receiver is disabled. TE and RE do not disable the bit clock or the frame sync generation. To disable the bit clock and the frame sync generation, the SSIEN bit in the SCR can be cleared or TFR_CLK_DIS/RFR_CLK_DIS bits can be set or the port control logic external to the SSI (for example, in the IOMUX) can be reconfigured.

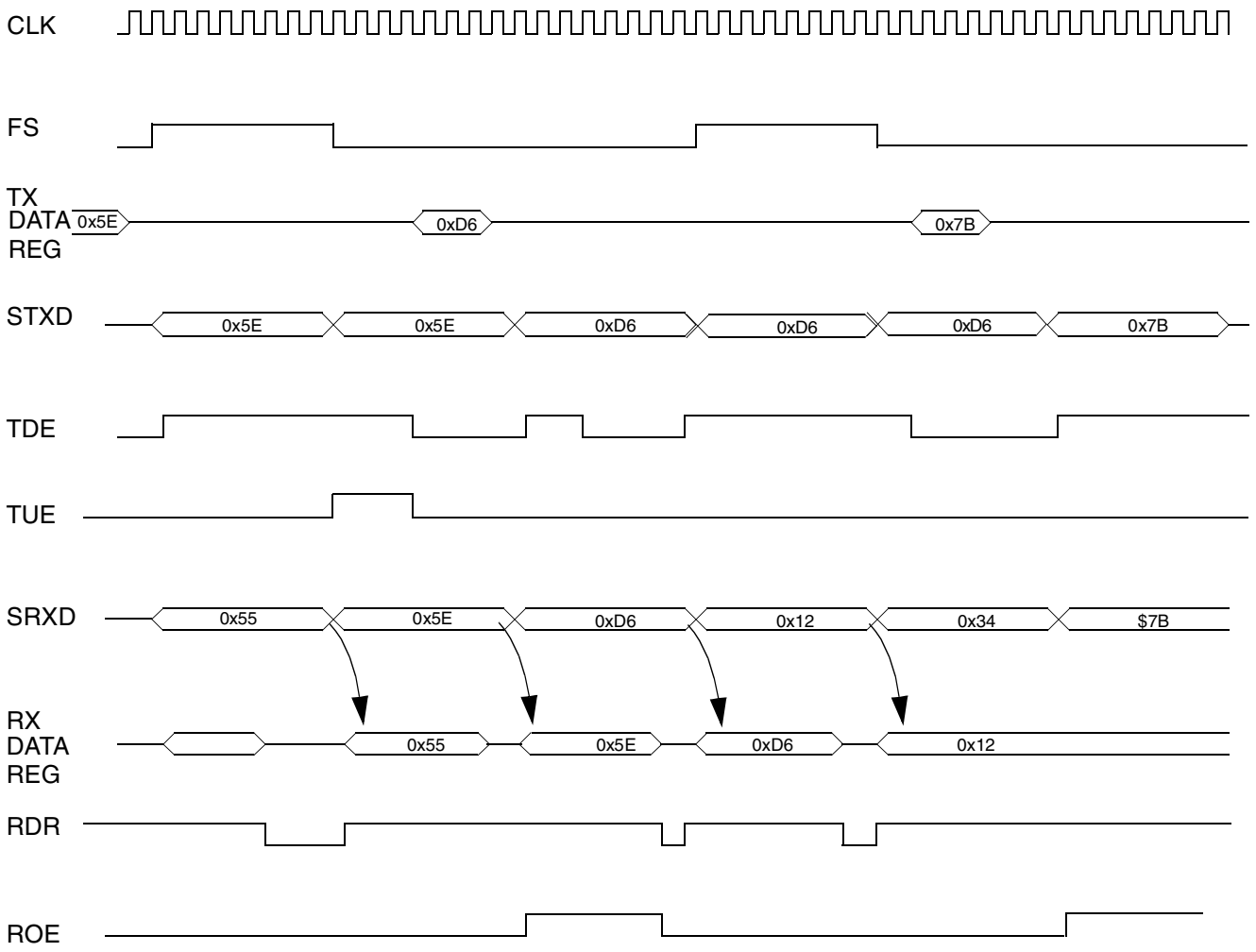
In the Two-Channel mode of operation, both the channels (Data Registers, FIFOs, Interrupts and DMA requests) operate in the same manner, as described above. The only difference in case of the second channel is that the Interrupts related to this channel are generated only in case this mode of operation is selected.

The transmitter and receiver timing for an 8-bit word with continuous clock, FIFO disabled, three words per frame sync in Network mode is shown in Figure 56-5 (“Network Mode Timing - Continuous Clock”).

NOTE

The transmitter repeats the value 0x5E because of an underrun condition

For the receive section, data received on the SRXD pin gets transferred to the Rx Data register at the end of each time slot. If the FIFO is disabled, the RDR flag gets set and causes a receiver interrupt if RE, RIE and RDR_EN bits are set. If the FIFO is enabled, then the RFF flag is used for interrupt generation (this flag is set in accordance with the watermark settings). Here all time slots are enabled. The receive data ready flag is set after reception of the first data (0x55). Since the flag is not cleared (Rx Data Register is not read by core), the Receive Overrun Error (ROE) flag is set on reception of the next data (0x5E). ROE flag is cleared on writing ‘1’ to the corresponding interrupt status bit in SSI Status Register.



Note: Processor must write '1' to the corresponding TUE/ROE Interrupt status bit in SISR to clear TUE/ROE Interrupt.

Figure 56-5. Network Mode Timing—Continuous Clock

56.1.2.3 Gated Clock Mode

Gated Clock mode is often used to hook up to SPI-type interfaces on Microcontroller Units (MCUs) or external peripheral chips. In Gated Clock mode, the presence of the clock indicates that valid data is on the STXD or SRXD ports. For this reason, no frame sync is needed in this mode. Once transmission of data has completed, the clock is pulled to the inactive state. Gated clocks are allowed for both the transmit and receive sections with either internal or external clock in Normal mode. Gated clocks are not allowed in Network mode. See Table 56-5 ("Clock Pin Configurations") for SSI configuration for gated-mode operation.

The clock runs when the TE bit and/or the RE bit are appropriately enabled. For the case of internally generated clock, all internal bit clocks, word clocks, and frame clocks continue to operate. When a valid

time slot occurs (such as the first time slot in Normal mode), the internal bit clock is enabled onto the appropriate clock port. This allows data to be transferred out in periodic intervals in Gated Clock mode. With an external clock, the SSI waits for a clock signal to be received. Once the clock begins, valid data is shifted in. Care should be taken to clear all DC bits (0x00000) when SSI is used in Gated mode. In gated mode of operation the TFS, RFS, TLS, RLS, TFRC, and RFRC bits of AISR register are not generated.

For Gated clock operated in external clock mode, a proper clock signalling must be applied to the SSI STCK in order for it to function properly. When TSCKP is 0, CLK_IST value should be 1. When TSCKP is 1, CLK_IST value should be 0. If the SSI uses rising edge transition to clock data (TSCKP=0) and the falling edge transition to latch data (RSCKP=0), the clock must be in an active low state when idle. If the SSI uses falling edge transition to clock data (TSCKP=1) and the rising edge transition to latch data (RSCKP=1), the clock must be in a active high state when idle. Figure 56-6 through Figure 56-9 illustrate the different edge clocking/latching.

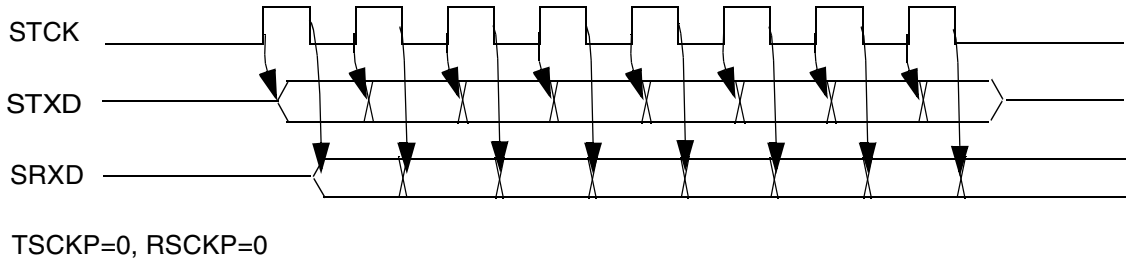


Figure 56-6. Internal Gated Mode Timing—Rising Edge Clocking/Falling Edge Latching

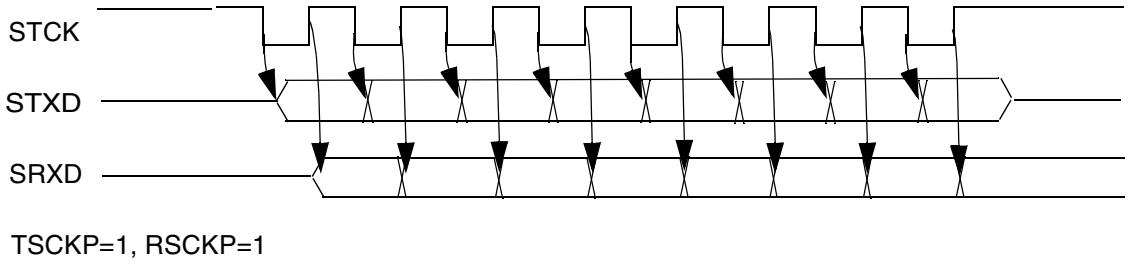


Figure 56-7. Internal Gated Mode Timing—Falling Edge Clocking/Rising Edge Latching

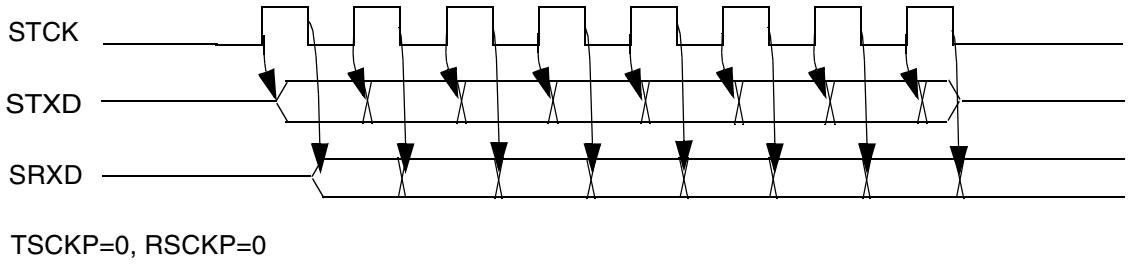


Figure 56-8. External Gated Mode Timing—Rising Edge Clocking/Falling Edge Latching

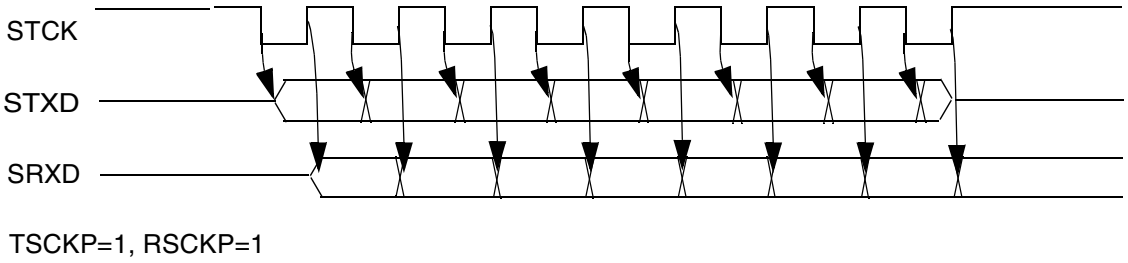


Figure 56-9. External Gated Mode Timing—Falling Edge Clocking/Rising Edge Latching

NOTE

- The bit clock ports must be kept free of timing glitches. If a single glitch occurs, all ensuing transfers will be out of synchronization.
- In case of External Gated Mode, even though the Tx Data line is put in the high-impedance state at the last non-active edge of the bit clock, the round trip delay should be sufficient to take care of hold time requirements at the external receiver.

56.1.2.4 I2S Mode

The SSI is compliant to I^2S bus specification from Philips Semiconductors (February 1986, Revised June 5, 1996). See Figure 56-10 for an illustration of the basic I2S protocol timing.

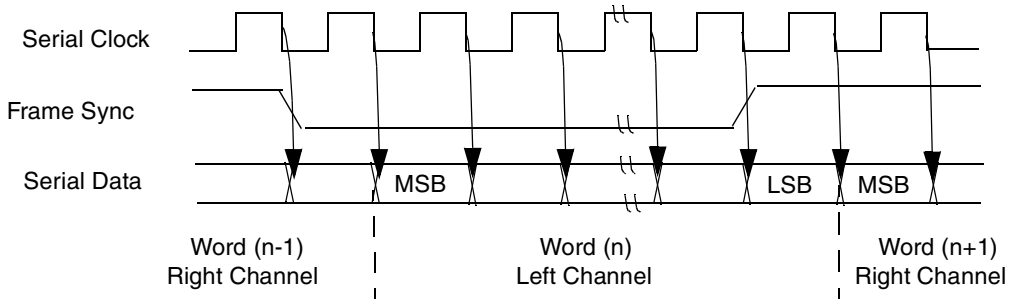


Figure 56-10. I^2S Mode Timing—Serial Clock, Frame Sync and Serial Data

Select I^2S mode using the options listed in Table 56-2.

Table 56-2. I^2S Mode Selection

I^2S_MODE[1]	I2S_MODE[0]	Mode Type
0	0	Normal mode
0	1	I2S master mode
1	0	I2S slave mode
1	1	Normal mode

In normal mode operation, no register bits are forced to any particular state internally and the user can program the SSI to work in any operating condition.

When I2S modes are entered (I2S master (01) or I2S slave (10)), the following settings are recommended:

- Sync mode (SCR[4] =1)
- Tx shift direction: MSB transmitted first (STCR[4]=0)
- Rx shift direction: MSB received first (SRCR[4]=0)
- Tx data clocked at falling edge of the clock (STCR[3]=1)
- Rx data latched at rising edge of the clock (SRCR[3]=1)
- Tx frame sync active low (STCR[2]=1)
- Rx frame sync active low (SRCR[2]=1)
- Tx frame sync initiated one bit before data is transmitted (STCR[0]=1)
- Rx frame sync initiated one bit before data is received (SRCR[0]=1)
- TX Frame Rate should be 2 i.e. (STCCR[12:8] = 1)
- RX Frame Rate should be 2 i.e. (SRCCR[12:8] = 1)

In I^2S master mode(SCR[6:5]=01), the following additional settings are recommended:

- TXDIR bit (STCR[5]) set to 1 to select internal generated bit clock
- TFDIR bit (STCR[6]) set to 1 to select internal generated frame sync

In I^2S master mode(SCR[6:5]=01), the following settings are internally overridden by the hardware:

- Network mode is selected (SCR[3]=1)
- Tx frame sync length set to one-word-long-frame (STCR[1]=0)
- Rx frame sync length set to one-word-long-frame (SRCR[1]=0)
- Tx shifting with regard to bit 0 of TXSR (STCR[9]=1)
- Rx shifting with regard to bit 0 of RXSR (SRCR[9]=1)

The user needs to set the following control bits to configure the bit clock and frame sync:

- PM (STCCR[7:0])
- PSR (STCCR[17])
- DIV2 (STCCR[18]
- WL (STCCR[16:13]
- DC (STCCR[12:8])

The word length is fixed to 32 in I^2S Master mode and the WL bits determine the number of bits that will contain valid data (out of the 32 transmitted/received bits in each channel). The fixing of word duration as 32 simplifies the relation between oversampling clock (ccm_ssi_clk) and Frame Sync (ccm_ssi_clk becomes an integer multiple of Frame Sync).

In I^2S slave mode (SCR[6:5]=10), the following additional settings are recommended:

- TXDIR bit (STCR[5]) set to 0 to select external generated bit clock
- TFDIR bit (STCR[6]) set to 0 to select external generated frame sync

In I^2S slave mode (SCR[6:5]=10), the following settings are internally overridden by the hardware:

- Normal mode is selected (SCR[3]=0)
- Tx frame sync length set to one-bit-long-frame (STCR[1]=1)
- Rx frame sync length set to one-bit-long-frame (SRCR[1]=1)
- Tx shifting with regard to bit 0 of TXSR (STCR[9]=1)
- Rx shifting with regard to bit 0 of RXSR (SRCR[9]=1)

The user needs to set the following control bits to configure the data transmission:

- WL (STCCR[16:13])
- DC (STCCR[12:8])

The word length is variable in I^2S slave mode and the WL bits determine the number of bits that will contain valid data. The actual word length is determined by the external CODEC. The external I^2S Master still sends frame sync according to the I^2S protocol (early, word wide and active low), the SSI internally operates so that each frame sync transition is the start of a new frame (the WL bits determine the number of bits to be transmitted/received). After one data word has been transferred, the SSI waits for the next frame sync transition to start operation in the next time slot. Transmit (STMSK) and receive (SRMSK) mask bits should not be used in I^2S Slave mode of operation.

56.1.2.5 AC97 Mode

In AC97 mode of operation, the SSI transmits a 16-bit Tag Slot at the start of a frame and the rest of the slots (in that frame) are all 20-bits wide. The same sequence is followed while receiving data. Refer to the AC97 specification for details regarding transmit and receive sequences and data formats.

Note that the SSI only has one RxDATA pin so the SSI can only support one codec. Secondary codecs are not supported.

When AC97 mode is enabled, the following settings are internally overridden by the hardware. The programmed register values are not changed by entering AC97 mode but they no longer apply to the module's operation. Writing to the programmed register fields will update their values; these updates can be seen by reading back the register fields. However, these settings will not take effect until AC97 mode is turned off.

The register bits within the bracket are the equivalent settings:

- Sync mode is entered (SCR[4] =1)
- Network mode is selected (SCR[3]=1)
- Tx shift direction is MSB transmitted first (STCR[4]=0)
- Rx shift direction is MSB received first (SRCR[4]=0)
- Tx data is clocked at rising edge of the clock (STCR[3]=0)
- Rx data is latched at falling edge of the clock (SRCR[3]=0)
- Tx frame sync is active high (STCR[2]=0)
- Rx frame sync is active high (SRCR[2]=0)
- Tx frame sync length is one-word-long-frame (STCR[1]=0)

- Rx frame sync length is one-word-long-frame (SRCR[1]=0)
- Tx frame sync initiated one bit before data is transmitted (STCR[0]=1)
- Rx frame sync initiated one bit before data is received (SRCR[0]=1)
- Tx shifting w.r.t. bit 0 of TXSR (STCR[9]=1)
- Rx shifting w.r.t. bit 0 of RXSR (SRCR[9]=1)
- Tx FIFO is enabled (STCR[7]=1)
- Rx FIFO is enabled (SRCR[7]=1)
- TFDIR bit (STCR[6]) is forced to 1 internally to select internal generated frame sync
- TXDIR bit (STCR[5]) is forced to 0 internally to select external generated bit clock

Any alteration of these bits individually will not affect the operational conditions of the SSI unless AC97 mode is deselected.

Hence, the only control bits needed to be set by the user to configure the data transmission/reception are the WL (STCCR[16:13]) and DC (STCCR[12:8]) bits. In AC97 mode, the WL bits can only legally take the values corresponding to 16-bit (truncated data) or 20-bit time slots. In case WL bits are set to select 16-bit time slots, the SSI pads the transmit data (four least significant bits) with zeros and while receiving, stores only the most significant 16 bits in the Rx FIFO.

Follow this sequence to program the SSI to work in AC97 mode:

1. Program the WL bits to a value corresponding to either 16 or 20 bits. The WL bit setting is only for the data portion of the AC97 frame (Slots #3 through #12). The Tag slot (Slot #0) is always 16 bits wide and the Command Address and Command Data slots (Slots #1 and #2) are always 20 bits wide.
2. Select the number of time slots by programming the DC bits. For AC97 operation, DC bits should be set to a value of 0xC, resulting in 13 time slots per frame.
3. Write data to be transmitted, in Tx FIFO 0 (through Tx Data Register 0)
4. Program the FV, TIF, RD, WR, and FRDIV bits in SACNT register
5. Update the contents of SACADD, SACDAT, and SATAG (for Fixed mode only) registers
6. Enable the AC97 mode of operation (AC97EN bit in SACNT register)

Once the SSI starts transmitting and receiving data (after being configured in AC97 mode), the programmer needs to service the interrupts, as and when they are raised (updates to command address/data or tag registers, reading of received data and writing more data for transmission). Further details regarding fixed and variable mode implementation are provided in the following sections.

While using AC97 in Two-Channel Mode (TCH_EN=1), it is recommended that the received tag is not stored in the Rx FIFO (TIF=0). In case the programmer needs to update the SATAG register and also issue a RD/WR command (in a single frame), it is recommended that the SATAG register be updated prior to issuing a RD/WR command.

56.1.2.5.1 AC97 Fixed Mode (SACNT[1]=0)

In fixed mode of operation, SSI transmits in accordance with the Frame Rate Divider bits which decides the number of frames for which the SSI should be idle, after operating for one frame.

In a valid frame, TAG Value (written by Core) will be transmitted in Slot #0, Command Address will be transmitted in Slot #1 in case of RD/WR Command, and Command Data will be transmitted in Slot #2 in case of a WR Command. The data from TX-FIFO is transmitted in Slot #3–Slot #12 depending on the valid slots indicated by the TAG value.

While receiving, bit 15 of the TAG Value (Slot #0) is checked to see if the CODEC is ready. If this bit is set, the frame is received. The received TAG provides the information about Slots containing valid data. The the corresponding TAG bit is valid, the Command Address (Slot #1) and Command Data (Slot #2) values are stored in the corresponding registers. The received data (Slot #3–Slot #12) is then stored in the Rx-FIFO (for valid slots).

56.1.2.5.2 AC97 Variable Mode (SACNT[1]=1)

In Variable Mode, the transmit slots which should contain data in the current frame are determined by SLOTREQ bits received in the previous frame. While receiving, if the CODEC is ready, the frame is received and the SLOTREQ bits (contained in Slot #1) are stored for scheduling transmission in the next frame.

The SACCST, SACCEN, and SACCDIS registers helps in determining which transmit slots are active. This information is used to ensure that SSI does not transmit data for powered-down/inactive channels.

56.1.2.6 External Frame and Clock Operation

When applying external frame sync and clock signals to SSI, there should be at least 4-bit clock cycles between the enabling of the transmit or receive section and the rising edge of the corresponding frame sync signal. The transition of STFS or SRFS should be synchronized with the rising edge of external clock signal, STCK or SRCK.

56.1.2.7 Data Alignment Formats Supported

The SSI supports three data formats in order to provide flexibility with handling data. These formats dictate how data is written to (and read from) the data registers. Therefore, data can appear in different places in STX0/1 and SRX0/1 based on the data format and the number of bits per word. Independent data formats are supported for both the transmitter and receiver (that is, the transmitter and receiver can use different data formats).

The supported data formats are:

- MSB alignment
- LSB alignment
  - — Zero-extended (receive data only)
  - — Sign-extended (receive data only)

With MSB alignment, the most significant byte is bits 31 through 24 of the data register if the word length is larger than or equal to 16 bits. If the word length is less than 16 bits and MSB alignment is chosen, the most significant byte is bits 15 through 8. With LSB alignment, the least significant byte is bits 7 through

0. Data alignment is controlled by the TXBIT0 bit in the STCR and the RXBIT0 bit in the SRCR. See Table 56-3 for the bit assignment for all the data formats supported by the SSI.

Table 56-3. Data Alignment

Format	Bit Number																													
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
8-bit LSB Aligned																									7	6	5	4	3	2	1	0
8-bit MSB Aligned																	7	6	5	4	3	2	1	0								
10-bit LSB Aligned																							9	8	7	6	5	4	3	2	1	0
10-bit MSB Aligned																	9	8	7	6	5	4	3	2	1	0						
12-bit LSB Aligned																					11	10	9	8	7	6	5	4	3	2	1	0
12-bit MSB Aligned																	11	10	9	8	7	6	5	4	3	2	1	0				
16-bit LSB Aligned																	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
16-bit MSB Aligned	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																
18-bit LSB Aligned															17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
18-bit MSB Aligned	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0														
20-bit LSB Aligned													19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
20-bit MSB Aligned	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0												
22-bit LSB Aligned											21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
22-bit MSB Aligned	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0										
24-bit LSB Aligned									23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
24-bit MSB Aligned	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								

In addition, receive data can either be zero-extended or sign-extended if LSB alignment is selected. With zero-extension, all bits above the most significant bit are 0s. This format is useful when data is stored in a pure integer format. With sign-extension, all bits above the most significant bit are equal to the most significant bit. This format is useful when data is stored in a fixed-point integer format (which implies fractional values). Receive data extension is controlled by the RXEXT bit in the SRCR. Transmit data used

with LSB alignment has no concept of sign/zero-extension. Unused bits above the most significant bit are simply ignored.

When configured in I2S or AC97 mode, the SSI forces the selection of LSB alignment. However, RXEXT still permits a choice between zero-extension and sign-extension.

Refer to Section 56.3.3.10, SSI Transmit Configuration Register (STCR)" and Section 56.3.3.11, SSI Receive Configuration Register (SRCR)" for more details on the relevant bits in the STCR and SRCR registers.

56.2 External Signal Description

The Synchronous Serial Interface (SSI) is connected to chip pads through the Digital Audio Mux (AUDMUX) module or directly as well. Refer to the AUDMUX chapter for programming details of various multiplexing options. Table 56-4 shows the signal properties.

Table 56-4. Signal Properties

Name	Port	I/O	Function	Reset State	Pull up
SRCK	—	I/O	Serial Receive Clock	0	Passive
SRFS	—	I/O	Serial Receive Frame Sync	0	Passive
SRXD	—	I	Serial Receive Data	—	—
STCK	—	I/O	Serial Transmit Clock	0	Passive
STFS	—	I/O	Serial Transmit Frame Sync	0	Passive
STXD	—	O	Serial Transmit Data	0	Passive

56.2.1 Detailed Signal Descriptions

This section provides detailed signal descriptions.

56.2.1.1 SRCK—Serial Receive Clock

The SRCK port can be used as either an input or an output. This clock signal is used by the receiver in asynchronous mode and is always continuos. During synchronous mode, the STCK port is used instead for clocking in data. In SSI synchronous modes, this port can be used as an output port for the oversampling clock, (ccm_ssi_clk). In I^2S master mode, this port can be used to output ccm_ssi_clk to external CODEC.

56.2.1.2 SRFS—Serial Receive Frame Sync

The SRFS port can be used as either an input or an output. The frame sync is used by the receiver to synchronize the transfer of data. The frame sync signal can be one bit or one word in length and can occur one bit before the transfer of data or right at the transfer of data. If SRFS is configured as input, the external device should drive SRFS during rising edge of STCK or SRCK.

56.2.1.3 SRXD—Serial Receive Data

The SRXD port is an input and is used to bring serial data into the Receive Data Shift Register.

56.2.1.4 STCK—Serial Transmit Clock

The STCK port can be used as either an input or an output. This clock signal is used by the transmitter and can be either continuous or gated. During Gated Clock mode, data on the STCK port is valid only during the transmission of data, otherwise it is pulled to the inactive state. In Synchronous mode, this port is used by both the transmit and receive sections.

56.2.1.5 STFS—Serial Transmit Frame Sync

The STFS port can be used as either an input or an output. The frame sync is used by the transmitter to synchronize the transfer of data. The frame sync signal can be one bit or one word in length and can occur one bit before the transfer of data or right at the transfer of data. In Synchronous mode, this port is used by both the transmit and receive sections. In Gated Clock mode, frame sync signals are not used. If STFS is configured as input, the external device should drive STFS during rising edge of STCK if TSCKP is +ve edge triggered. The external device should drive STFS during falling edge of STCK if TSCKP is –ve edge triggered.

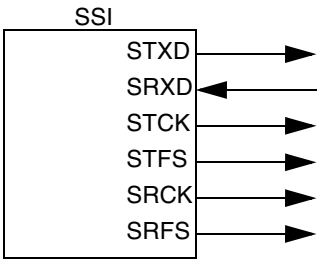
56.2.1.6 STXD—Serial Transmit Data

The STXD port is an output and transmits data from the Serial Transmit Shift Register. The STXD port is an output port when data is being transmitted and is disabled between data word transmissions and on the trailing edge of the bit clock after the last bit of a word is transmitted.

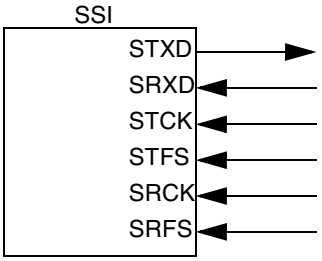
Figure 56-11 (“Asynchronous (SYN=0) SSI Configurations—Continuous Clock”) and Figure 56-12 (“Synchronous SSI Configurations—Continuous and Gated Clock”) show the main SSI configurations. These ports support all transmit and receive functions with continuous or gated clock as shown.

NOTE

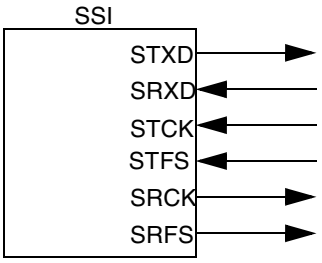
Gated clock implementations do not require the use of the frame sync ports (STFS and SRFS).



SSI Internal Continuous Clock for TX/RX (RXDIR=1,TXDIR=1,RFDIR=1,TFDIR=1,SYN=0)

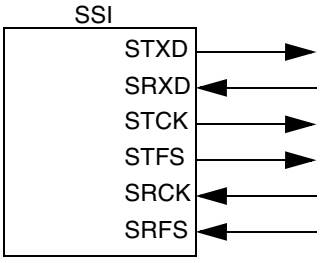


SSI External Continuous Clock for TX/RX (RXDIR=0,TXDIR=0,RFDIR=0,TFDIR=0,SYN=0)



SSI Internal Continuous Clock for RX (RXDIR=1, TXDIR=0,RFDIR=1,TFDIR=0, SYN=0)

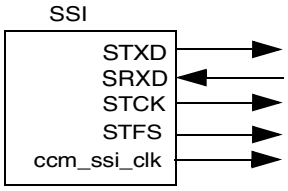
SSI External Continuous Clock for TX



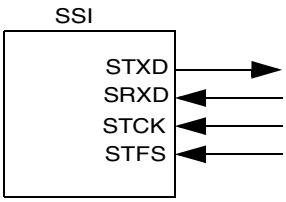
SSI Internal Continuous Clock for TX (RXDIR=0, TXDIR=1, RFDIR=0, TFDIR=1, SYN=0)

SSI EXternal Continuous Clock for RX

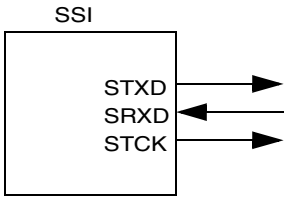
Figure 56-11. Asynchronous (SYN=0) SSI Configurations—Continuous Clock



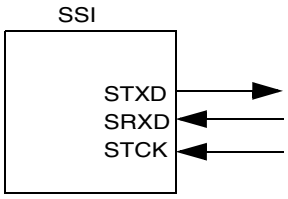
SSI Internal Continuous Clock (RXDIR=0, TXDIR=1, RFDIR=X, TFDIR=1, SYN=1, SYS_CLK_EN = 1)
SSI I2S Master Mode(I2S_Mode=01, SYS_CLK_EN)



SSI External Continuous Clock (RXDIR=0, TXDIR=0, RFDIR=X, TFDIR=0,SYN=1)
SSI I2S Slave Mode(I2S_Mode=10)



SSI Internal Gated Clock (RXDIR=1, TXDIR=1, SYN=1)



SSI External Gated Clock (RXDIR=1, TXDIR=0, SYN=1)

Figure 56-12. Synchronous SSI Configurations—Continuous and Gated Clock

See Figure 56-13 for an example of the port signals for an 8-bit data transfer. Continuous and gated clock signals are shown, as well as the bit-length frame sync signal and the word-length frame sync signal.

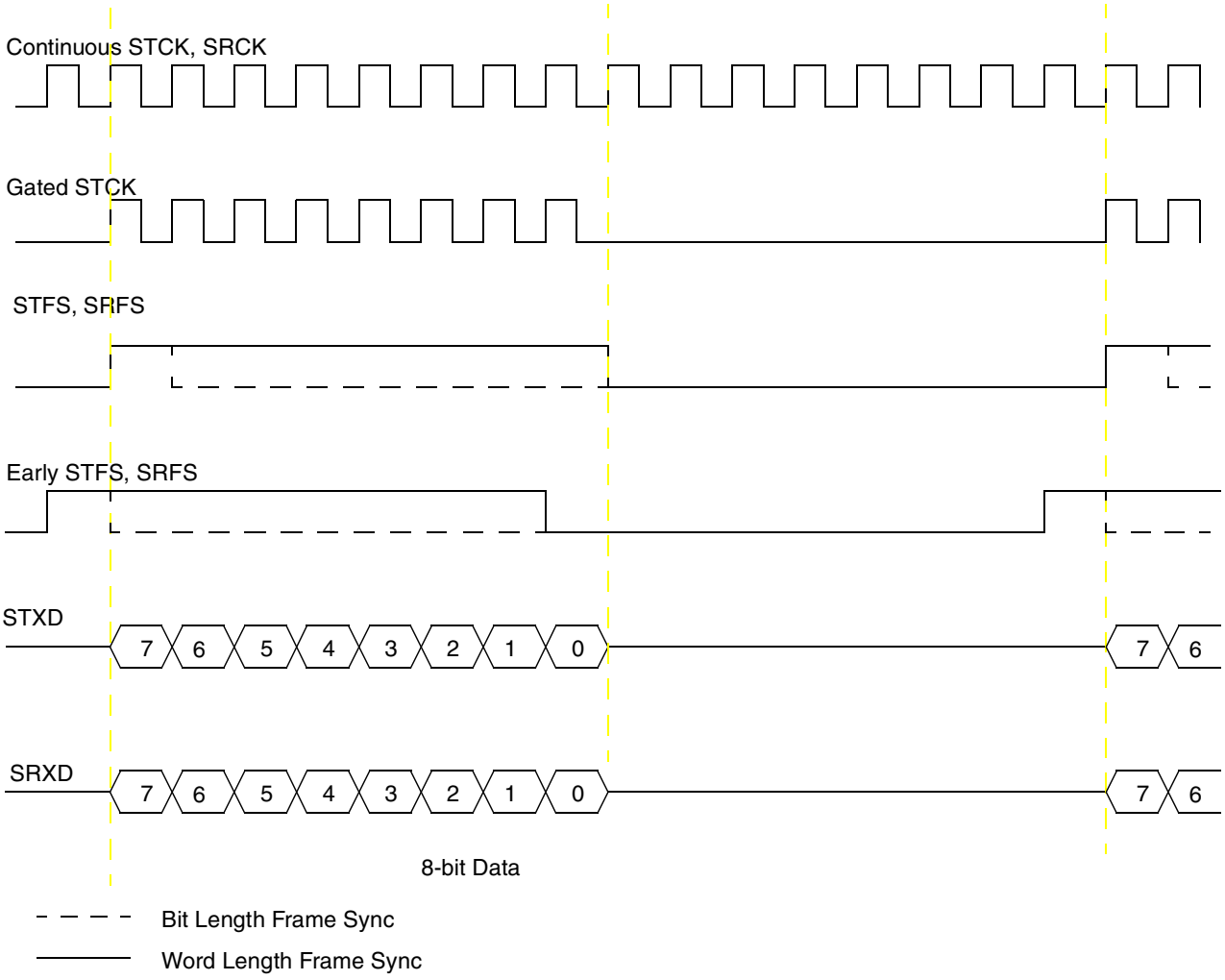


Figure 56-13. Serial Clock and Frame Sync Timing

Table 56-5 lists the clock pin configurations.

Table 56-5. Clock Pin Configurations

SYN	RXDIR	TXDIR	RFDIR	TFDIR	SRCK	STCK	SRFS	STFS
Asynchronous Mode								
0	0	0	0	0	RCK in	TCK in	RFS in	TFS in
0	0	0	0	1	RCK in	TCK in	RFS in	TFS out
0	0	0	1	0	RCK in	TCK in	RFS out	TFS in
0	0	0	1	1	RCK in	TCK in	RFS out	TFS out
0	0	1	0	0	RCK in	TCK out	RFS in	TFS in

Table 56-5. Clock Pin Configurations

SYN	RXDIR	TXDIR	RFDIR	TFDIR	SRCK	STCK	SRFS	STFS
0	0	1	0	1	RCK in	TCK out	RFS in	TFS out
0	0	1	1	0	RCK in	TCK out	RFS out	TFS in
0	0	1	1	1	RCK in	TCK out	RFS out	TFS out
0	1	0	0	0	RCK out	TCK in	RFS in	TFS in
0	1	0	0	1	RCK out	TCK in	RFS in	TFS out
0	1	0	1	0	RCK out	TCK in	RFS out	TFS in
0	1	0	1	1	RCK out	TCK in	RFS out	TFS out
0	1	1	0	0	RCK out	TCK out	RFS in	TFS in
0	1	1	0	1	RCK out	TCK out	RFS in	TFS out
0	1	1	1	0	RCK out	TCK out	RFS out	TFS in
0	1	1	1	1	RCK out	TCK out	RFS out	TFS out
Synchronous Mode								
1	0	0	x	0	—	CK in	—	FS in
1	0	0	x	1	—	CK in	—	FS out
1	0	1	x	0	—	CK out	—	FS in
1	0	1	x	1	—	CK out	—	FS out
1	1	0	x	x	—	Gated in	—	—
1	1	1	x	x	—	Gated out	—	—

56.3 Memory Map and Register Definition

Section 56.3.3, Register Descriptions” provides the detailed descriptions for all of the SSI registers.

56.3.1 SSI Memory Map

Table 56-7 shows the SSI memory map.

Table 56-7. SSI Memory Map

Address	Register	Access	Reset Value	Section/Page
0x43FA+0x0000 (STX0_)	SSI Transmit Data Register	R/W	0x0000_0000	56.3.3.1/56-32
0x43FA+0x0004 (STX1_)	SSI Transmit Data Register 1	R/W	0x0000_0000	

Table 56-7. SSI Memory Map (continued)

0x43FA+0x0008 (SRX0_)	SSI Receive Data Register 0	Read-only	0x0000_0000	56.3.3.4/56-36
0x43FA+0x000C (SRX1_)	SSI Receive Data Register 1	Read-only	0x0000_0000	
0x43FA+0x0010 (SCR)	SSI Control Register	R/W	0x0000_0000	56.3.3.7/56-39
0x43FA+0x0014 (SISR)	SSI Interrupt Status Register	Read-only	0x0000_3003	56.3.3.8/56-41
0x43FA+0x0018 (SIER)	SSI Interrupt Enable Register	R/W	0x0000_3003	56.3.3.9/56-46
0x43FA+0x001C (STCR)	SSI Transmit Configuration Register	R/W	0x0000_0200	56.3.3.10/56-47
0x43FA+0x0020 (SRCR)	SSI Receive Configuration Register	R/W	0x0000_0200	56.3.3.11/56-49
0x43FA+0x0024 (STCCR)	SSI Transmit Clock Control Register	R/W	0x0004_0000	56.3.3.12/56-51
0x43FA+0x0028 (SRCCR)	SSI Receive Clock Control Register	R/W	0x0004_0000	
0x43FA+0x002C (SFCSR)	SSI FIFO Control/Status Register	R/W	0x0081_0081	56.3.3.13/56-53
0x43FA+0x0030 (STR)	SSI Test Register[1]	R/W	0x0000_1111	56.3.3.14/56-58
0x43FA+0x0034 (SOR)	SSI Option Register[2]	R/W	0x0000_0000	56.3.3.15/56-59
0x43FA+0x0038 (SACNT)	SSI AC97 Control Register	R/W	0x0000_0000	56.3.3.16/56-60
0x43FA+0x003C (SACADD)	SSI AC97 Command Address Register	R/W	0x0000_0000	56.3.3.17/56-62
0x43FA+0x0040 (SACDAT)	SSI AC97 Command Data Register	R/W	0x0000_0000	56.3.3.18/56-63
0x43FA+0x0044 (SATAG)	SSI AC97 Tag Register	R/W	0x0000_0000	56.3.3.19/56-64
0x43FA+0x0048 (STMSK)	SSI Transmit Time Slot Mask Register	R/W	0x0000_0000	56.3.3.20/56-65
0x43FA+0x004C (SRMSK)	SSI Receive Time Slot Mask Register	R/W	0x0000_0000	56.3.3.21/56-66
0x43FA+0x0050 (SACCST)	SSI AC97 Channel Status Register	R	0x0000_0000	56.3.3.22/56-67
0x43FA+0x0054 (SACCEN)	SSI AC97 Channel Enable Register	W	0x0000_0000	56.3.3.23/56-68
0x43FA+0x0058 (SACCDIS)	SSI AC97 Channel Disable Register	W	0x0000_0000	56.3.3.24/56-69

[1]SSI Test Register is intended for debugging purposes only and is not visible to the end user.
[2]SSI Option Register intended for internal use only and is not visible to the end user.

56.3.2 Register Summary

Figure 56-14 shows the key to the register fields and Table 56-8 shows the register figure conventions.

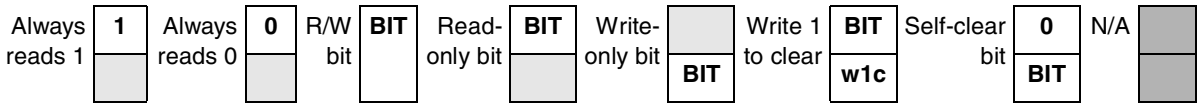


Figure 56-14. Key to Register Fields

Table 56-8. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 56-9 shows the SSI register summary.

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43FA+0x0000 (STX0_)	R	STX0[31:16]															
	W																
	R	STX0[15:0]															
	W																
0x43FA+0x0004 (STX1_)	R	STX1[31:16]															
	W																
	R	STX1[15:0]															
	W																
0x43FA+0x0008 (SRX0_)	R	SRX0[31:16]															
	W																
	R	SRX0[15:0]															
	W																
0x43FA+0x000C (SRX1_)	R	SRX1[31:16]															
	W																
	R	SRX1[15:0]															
	W																
0x43FA+0x0010 (SCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	RFR_CLK_DIS	TFR_CLK_DIS	CLK_IST	TCH_EN	SYS_CLK_EN	I2S MODE[1:0]		SYN	NET	RE	TE	SSI EN
	W																
0x43FA+0x0014 (SISR)	R	0	0	0	0	0	0	0	RFRC	TFRC	0	0	0	0	CMDAU	CMDDU	RXT
	W																
	R	RDR1	RDR0	TDE1	TDE0	ROE1	ROE0	TUE1	TUE0	TFS	RFS	TLS	RLS	RFF1	RFF0	TFE1	TFE0
	W																

Table 56-9. SSI Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43FA+0x0018 (SIER)	R	0	0	0	0	0	0	0	RFRC_EN	TFRC_EN	RDMAE	RIE	TDMAE	TIE	CMDAU_EN	CMDDU_EN	RXT_EN
	W																
	R	RDR1_EN	RDR0_EN	TDE1_EN	TDE0_EN	ROE1_EN	ROE0_EN	TUE1_EN	TUE0_EN	TFS_EN	RFS_EN	TLS_EN	RLS_EN	RFF1_EN	RFF0_EN	TFE1_EN	TFE0_EN
	W																
0x43FA+0x001C (STCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	TXBIT0	TFEN1	TFEN0	TFDIR	TXDIR	TSHFD	TSCKP	TFSI	TFSL	TEFS
	W																
0x43FA+0x0020 (SRCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	RXEXT	RXBIT0	RFEN1	RFEN0	RFDIR	RXDIR	RSHFD	RSCKP	RFSI	RFSL	REFS
	W																
0x43FA+0x0024 (STCCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	DIV2	PSR	WL3
	W																
	R	WL2	WL1	WL0	DC4	DC3	DC2	DC1	DC0	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
	W																
0x43FA+0x0028 (SRCCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	DIV2	PSR	WL3
	W																
	R	WL2	WL1	WL0	DC4	DC3	DC2	DC1	DC0	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
	W																
0x43FA+0x002C (SFCSR)	R	RFCNT1[3:0]				TFCNT1[3:0]				RFWM1[3:0]				TFWM1[3:0]			
	W																
	R	RFCNT0[3:0]				TFCNT0[3:0]				RFWM0[3:0]				TFWM0[3:0]			
	W																
0x43FA+0x0030 (STR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	TEST	RCK2TCK	RFS2TFS	RXSTATE[4:0]					TXD2RXD	TCK2RCK	TFS2RFS	TXSTATE[4:0]				
	W																

Table 56-9. SSI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43FA+0x0034 (SOR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	CLK OFF	RX_ CLR	TX_ CLR	INIT	WAIT[1:0]		SYN RST
	W																
0x43FA+0x0038 (SACNT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	FRDIV[5:0]						WR	RD	TIF	FV	AC97EN
	W																
0x43FA+0x003C (SACADD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	SACADD[18:16]		
	W																
	R	SACADD[15:0]															
	W																
0x43FA+0x0040 (SACDAT)	R	0	0	0	0	0	0	0	0	0	0	0	0	SACDAT[19:16]			
	W																
	R	SACDAT[15:0]															
	W																
0x43FA+0x0044 (SATAG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	SATAG[15:0]															
	W																

Table 56-9. SSI Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x43FA+0x0048 (STMSK)	R	STMSK[31:0]															
	W																
	R																
	W																
0x43FA+0x004C (SRMSK)	R	SRMSK[31:0]															
	W																
	R																
	W																
0x43FA+0x0050 (SACCST)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	SACCST[9:0]									
	W																
0x43FA+0x0054 (SACCEN)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W							SACCEN[9:0]									
0x43FA+0x0058 (SACCDIS)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W							SACCDIS[9:0]									
	W																

Table 56-9. SSI Register Summary (continued)

56.3.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams in bit order.

56.3.3.1 SSI Transmit Data Registers 0 & 1 (STX0/1)

See Figure 56-15 for illustration of valid bits in the SSI0 Transmit Data Register and Table 56-10 for description of the bit fields in the register.

0x43FA+0x0000 (STX0_) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STX0[31:16]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STX0[15:0]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-15. SSI0 Transmit Data Register

See Figure 56-16 for illustration of valid bits in SSI1 Transmit Data Register and Table 56-10 for description of the bit fields in the register.

0x43FA+0x0004 (STX1_) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STX1[31:16]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STX1[15:0]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-16. SSI1 Transmit Data Register

Table 56-10. SSI Transmit Data Register Field Descriptions

Field	Description
31–0 STX0 STX1	SSI Transmit Data. These bits store the data to be transmitted by the SSI. These are implemented as the first word of their respective Tx FIFOs. Data written to these registers is transferred to the Transmit Shift Register (TXSR), when shifting of the previous data is complete. If both FIFOs are in use, data is alternately transferred from STX0 and STX1, to TXSR. Multiple writes to the STX registers will not result in the previous data being over-written by the subsequent data. STX1 can only be used in Two-Channel mode of operation. Protection from over-writing is present irrespective of whether the transmitter is enabled or not. Example 1: If Tx FIFO0 is in use and user writes Data1...Data16 to STX0, Data16 will not over-write Data1. Data1...Data15 are stored in the FIFO while Data16 is discarded. Example 2: If Tx FIFO0 is not in use and user writes Data1, Data2 to STX0, then Data2 will not over-write Data1 and will be discarded.

NOTE

Enable SSI (SSIEN=1) before writing to SSI Transmit Data Registers.

56.3.3.2 SSI Transmit FIFO 0 and 1 Registers

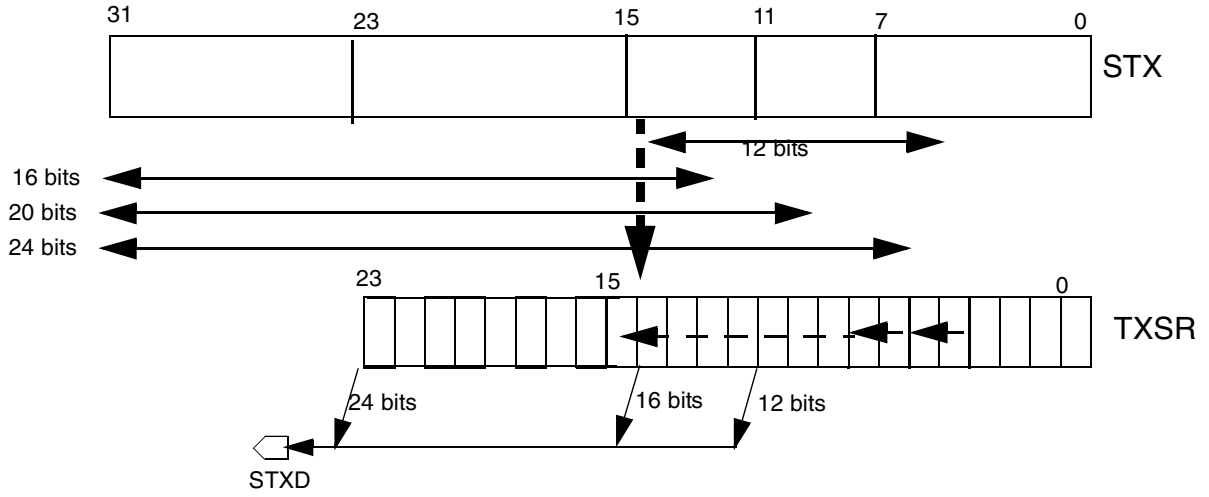
The SSI Transmit FIFO registers are 15 × 32-bit registers. These registers are not directly accessible by the end user. Transmit Shift Register (TXSR) receives its values from these FIFO registers. Transmitted data is first-in-first-out. When the Transmit Interrupt Enable (TIE) bit in the SIER and either of the Transmit FIFO Empty Enable (TFE0 or 1) bits in the SIER are set, the core is interrupted whenever the number of empty slots exceed or are equal to the selected threshold value of corresponding Tx-FIFO.

56.3.3.3 SSI Transmit Shift Register (TXSR)

The SSI Transmit Shift Register (TXSR) is a 24-bit shift register that contains the data being transmitted. This register is not directly accessible by the end user. When a continuous clock is used, data is shifted out to the Serial Transmit Data (STXD) port by the selected (internal/external) bit clock when the associated (internal/external) frame sync is asserted. When a gated clock is used, data is shifted out to the STXD port by the selected (internal/external) gated clock. The Word Length control bits (WL[3:0]) in the STCCR (described in SSI Transmit and Receive Clock Control Registers) determine the number of bits to be shifted out of the TXSR before it is considered empty and can be written to again. This word length can be 8, 10, 12, 16, 18, 20, 22 or 24 bits. The data to be transmitted occupies the most significant portion of the shift register if TXBIT0 is ‘0’, otherwise it occupies the least significant portion. The unused portion of the register is ignored. Data is always shifted out of this register with the Most Significant Bit (MSB) first when the SHFD bit of the STCR is cleared. If this bit is set, the Least Significant Bit (LSB) is shifted out first.

Figure 56-17 through Figure 56-20 show the transmitter loading and shifting operation. The figures show the working for some WL values, the same can be extended for other values.

Figure 56-17. Transmit Data Path (TXBIT0=0, TSHFD=0) (MSB Alignment)



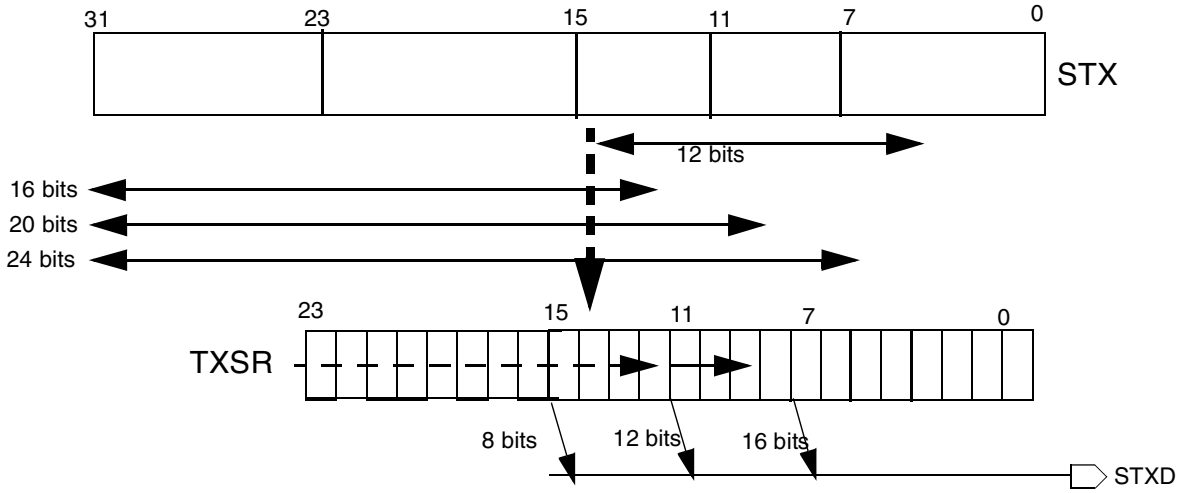


Figure 56-18. Transmit Data Path (TXBIT0=0, TSHFD=1) (MSB Alignment)

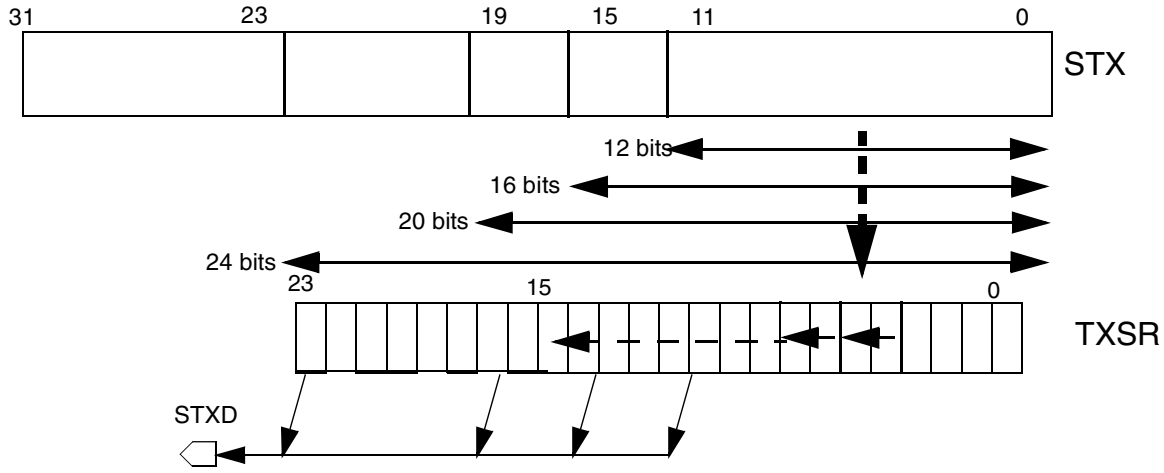


Figure 56-19. Transmit Data Path (TXBIT0=1, TSHFD=0) (LSB Alignment)

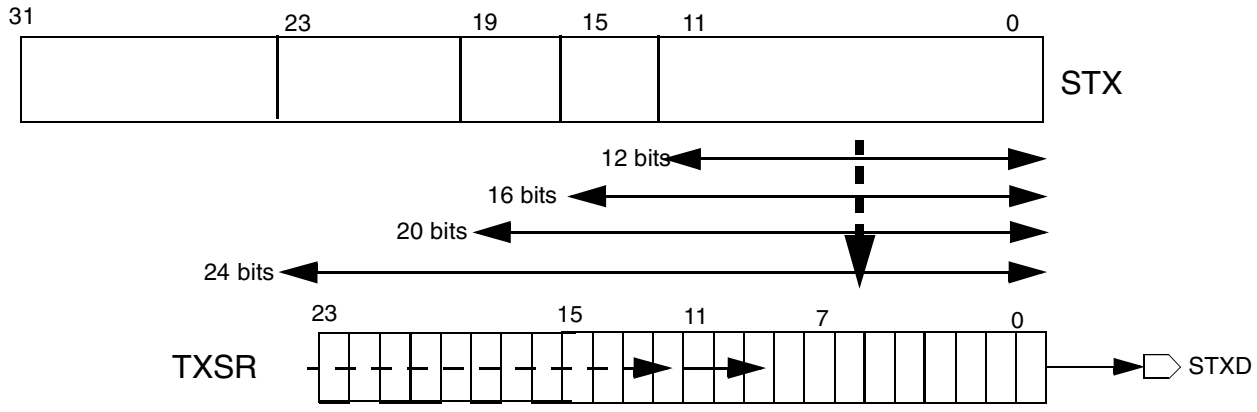


Figure 56-20. Transmit Data Path (TXBIT0=1, TSHFD=1) (LSB Alignment)

56.3.3.4 SSI Receive Data Registers 0 and 1 (SRX0/1)

See Figure 56-21 for illustration of valid bits in SSI0 Receive Data Register and Table 56-11 for description of the bit fields in the register.

0x43FA+0x0008 (SRX0_) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SRX0[31:16]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRX0[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-21. SSI0 Receive Data Register

See Figure 56-22 for illustration of valid bits in SSI1 Receive Data Register and Table 56-11 for description of the bit fields in the register.

0x43FA+0x000C (SRX1_) Access: User read-only
0x5001_40

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SRX1[31:16]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRX1[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-22. SSI1 Receive Data Register

Table 56-11. SSI_1 Receive Data Register Field Descriptions

Field	Description
31–0 SRX0 SRX1	SSI Receive Data. These bits store the data received by the SSI. These are implemented as the first word of their respective Rx FIFOs. These bits receive data from the RXSR depending on the mode of operation. In case both FIFOs are in use, data is transferred to each data register alternately. SRX1 can only be used in Two-Channel mode of operation.

56.3.3.5 SSI Receive FIFO 0 and 1 Registers

The SSI Receive FIFO Registers are 15 × 32-bit registers. These registers are not directly accessible by the end user (except in SSI test mode). They always accepts data from the Receive Shift Register (RXSR). The core is interrupted when the data level in either of the SSI Receive FIFOs reaches the selected threshold, if the associated interrupt is enabled.

56.3.3.6 SSI Receive Shift Register (RXSR)

The SSI Receive Shift Register (RXSR) is a 24-bit, shift register that receives incoming data from the serial receive data SRXD port. This register is not directly accessible by the end user. When a continuous clock is used, data is shifted in by the selected (internal/external) bit clock when the associated (internal/external) frame sync is asserted. When a gated clock is used, data is shifted in by the selected (internal/external) gated clock. Data is assumed to be received MSB first if the SHFD bit of the SRCR is cleared. If this bit is set, the data is received LSB first. Data is transferred to the appropriate SSI Receive Data Register (SRX0/1) or Receive FIFOs (if the receive FIFO is enabled and the corresponding SRX is full) after 8, 10, 12, 16, 18, 20, 22 or 24 bits have been shifted in depending on the WL[3:0] control bits. For receiving less than 24 bits of data, LSB bits are appended with zero. The following figures show the receiver loading and shifting operation.

Figure 56-23–Figure 56-26 show the working for some WL values; the same can be extended for other values.

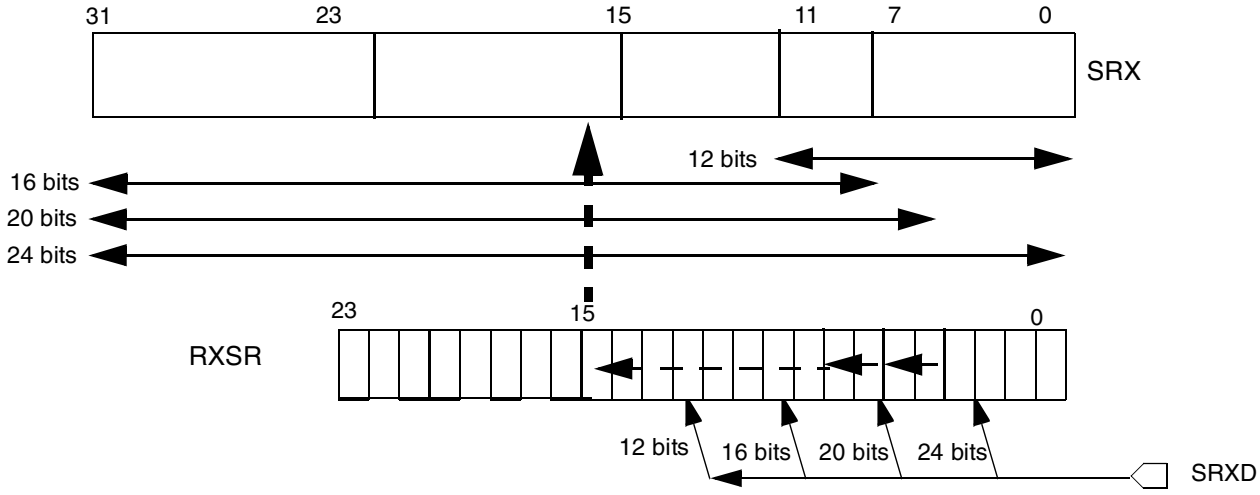


Figure 56-23. Receive Data Path (RXBIT0=0, RSHFD=0) (MSB Alignment)

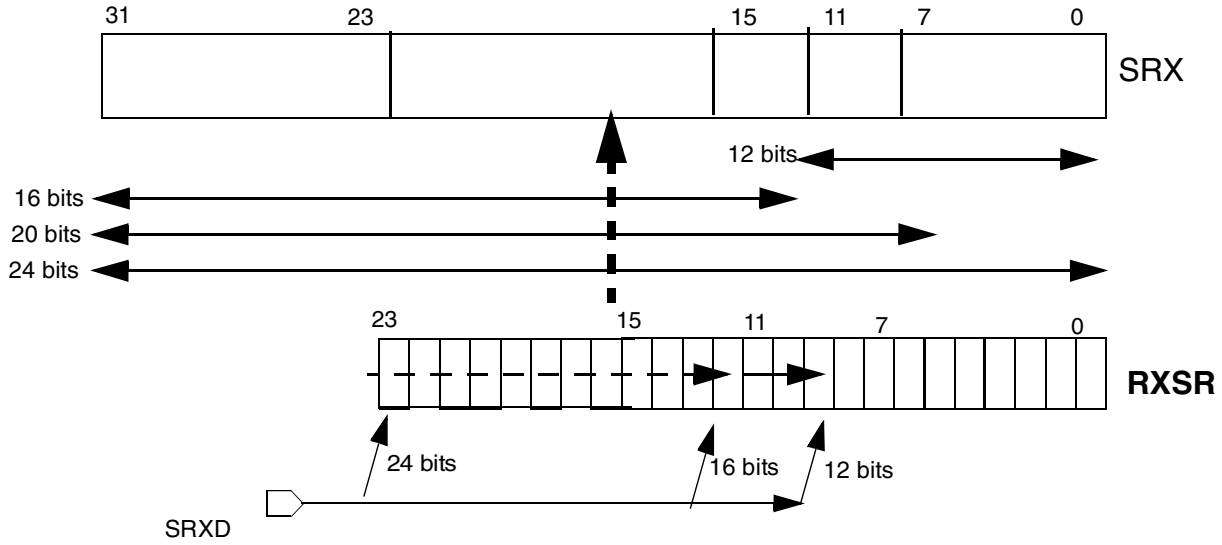


Figure 56-24. Receive Data Path (RXBIT0=0, RSHFD=1) (MSB Alignment)

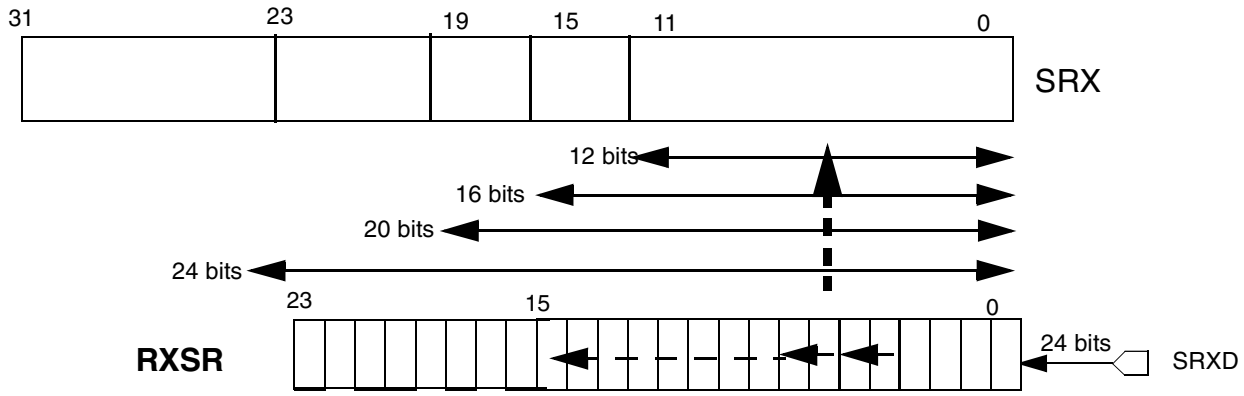


Figure 56-25. Receive Data Path (RXBIT0=1, RSHFD=0) (LSB Alignment)

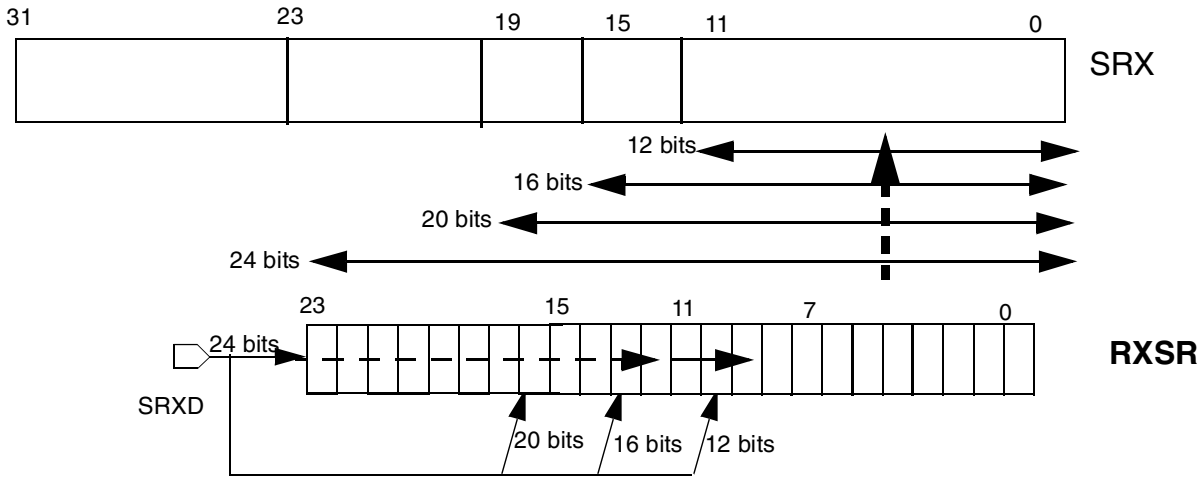


Figure 56-26. Receive Data Path (RXBIT0=1, RSHFD=1) (LSB Alignment)

56.3.3.7 SSI Control Register (SCR)

The SSI Control Register (SCR) is a 10-bit register used to set up the SSI. SSI reset is controlled by bit 0 in the SCR. SSI operating modes are also selected in this register (except AC97 mode which is selected in SACNT register).

See Figure 56-27 for illustration of valid bits in SSI Control Register and Table 56-12 for description of the bit fields in the register.

0x43FA+0x0010 (SCR)
0x5001_40

Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	RFR_CLK_DIS	TFR_CLK_DIS	CLK_IST	TCH_EN	SYS_CLK_EN	I2S MODE[1:0]		SYN	NET	RE	TE	SSIEN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-27. SSI Control Register

Table 56-12. SSI Control Register Field Descriptions

Field	Description
31–12	Reserved
11 RFR_CLK_DIS	Receive Frame Clock Disable. This bit provide option to keep the Frame-sync and Clock enabled or disabled after current receive frame, in which receiver is disabled by clearing RE bit. Writing to this bit has effect only when RE is disabled. 0 Continue Frame-sync/Clock generation after current frame during which RE is cleared. This may be required when Frame-sync and Clocks are required from SSI, even when no data is to be received. 1 Stop Frame-sync/Clock generation at next frame boundary. This will be effective also in case where receiver is already disabled in current or previous frames.
10 TFR_CLK_DIS	Transmit Frame Clock Disable. This bit provide option to keep the Frame-sync and Clock enabled or disabled after current transmit frame, in which transmitter is disabled by clearing TE bit. Writing to this bit has effect only when SSI is enabled TE is disabled. 0 Continue Frame-sync/Clock generation after current frame during which TE is cleared. This may be required when Frame-sync and Clocks are required from SSI, even when no data is to be received. 1 Stop Frame-sync/Clock generation at next frame boundary. This will be effective also in case where transmitter is already disabled in current or previous frames.
9 CLK_IST	Clock Idle State. This bit controls the idle state of the transmit clock port during SSI internal gated mode. Note: When Clock idle state is ‘1’ the clock polarity should always be negative edge triggered and when Clock idle = ‘0’ the clock polarity should always be positive edge triggered. 0 Clock idle state is ‘0’. 1 Clock idle state is ‘1’.

Table 56-12. SSI Control Register Field Descriptions (continued)

Field	Description
8 TCH_EN	Two-Channel Operation Enable. This bit allows SSI to operate in the two-channel mode. In this mode while receiving, the RXSR transfers data to SRX0 and SRX1 alternately and while transmitting, data is alternately transferred from STX0 and STX1 to TXSR. For an even number of slots, Two-Channel Operation can be enabled to optimize usage of both FIFOs or disabled as in the case of odd number of active slots. This feature is especially useful in I2S mode, where data for Left Speaker can be placed in Tx-FIFO0 and for Right speaker in Tx-FIFO1. 0 Two-channel mode disabled. 1 Two-channel mode enabled.
7 SYS_CLK_EN	System Clock (Oversampling Clock) Enable. When set, this bit allows the SSI to output the (ccm_ssi_clk) at the SRCK port, provided that synchronous mode, and transmit internal clock mode are set. The relationship between bit clock and ccm_ssi_clk is determined by DIV2, PSR, and PM bits. This feature is especially useful in I2S Master mode to output oversampling clock on SRCK port. 0 ccm_ssi_clk not output on SRCK port. 1 ccm_ssi_clk output on SRCK port.
6–5 I2S MODE[1:0]	I2S Mode Select. These bits allow the SSI to operate in Normal, I2S Master or I2S Slave mode. Refer to Section 56.1.2.4 for a detailed description of I2S Mode of operation. Refer to Table 56-2 (“Mode Selection”) for details regarding settings.
4 SYN	Synchronous Mode. This bit controls whether SSI is in synchronous mode or not. In synchronous mode, the transmit and receive sections of SSI share a common clock port (STCK) and frame sync port (STFS). 0 Asynchronous mode selected. 1 Synchronous mode selected.
3 NET	Network Mode. This bit controls whether SSI is in network mode or not. 0 Network mode not selected. 1 Network mode selected.
2 RE	Receive Enable. This control bit enables the receive section of the SSI. When this bit is enabled, data reception starts with the arrival of the next frame sync. If data is being received when this bit is cleared, data reception continues until the end of the current frame and then stops. If this bit is set again before the second to last bit of the last time slot in the current frame, then reception continues without interruption. 0 Receive section disabled. 1 Receive section enabled.

Table 56-12. SSI Control Register Field Descriptions (continued)

Field	Description
1 TE	Transmit Enable. This control bit enables the transmit section of the SSI. It enables the transfer of the contents of the STX registers to the TXSR and also enables the internal transmit clock. The transmit section is enabled when this bit is set and a frame boundary is detected. When this bit is cleared, the transmitter continues to send data until the end of the current frame and then stops. Data can be written to the STX registers with the TE bit cleared (the corresponding TDE bit will be cleared). If the TE bit is cleared and then set again before the second to last bit of the last time slot in the current frame, data transmission continues without interruption. The normal transmit enable sequence is to write data to the STX register(s) and then set the TE bit. The normal disable sequence is to clear the TE and TIE bits after the TDE bit is set. In gated clock mode, clearing the TE bit results in the clock stopping after the data currently in TXSR has shifted out. When the TE bit is set, the clock starts immediately (for internal gated clock mode). After enabling/disabling transmission, SSI expects 4 setup clock cycles before arrival of frame-sync for frame-sync to be accepted by SSI. In case of fewer clock cycles, there is high probability of the frame-sync to get missed. 0 Transmit section disabled. 1 Transmit section enabled.
0 SSIEN	SSIEN — SSI Enable This bit is used to enable/disable the SSI. When disabled, all SSI status bits are preset to the same state produced by the power-on reset, all control bits are unaffected, the contents of Tx and Rx FIFOs are cleared. When SSI is disabled, all internal clocks are disabled (except register access clock). 0 SSI is disabled. 1 SSI is enabled.

56.3.3.8 SSI Interrupt Status Register (SISR)

The SSI Interrupt Status Register (SISR) is used to monitor the SSI. This register is used by the core to interrogate the status of the SSI. In gated mode of operation the TFS, RFS, TLS, RLS, TFRC and RFRC bits of AISR register are not generated. The status bits are described in the following table.

NOTE

- SSI Status flags are valid when SSI is enabled.
- Refer to Section 56.4.3 and Section 56.4.4 for interrupt source mapping.
- All the flags in the SISR are updated after the first bit of the next SSI word has completed transmission or reception. Certain status bits (ROE0/1 and TUE0/1) are cleared by writing 1 to the corresponding interrupt status bit in SISR.

See Figure 56-28 for illustration of valid bits in SSI Interrupt Register and Table 56-13 for description of the bit fields in the register.

0x43FA+0x0014 (SISR) Access: User read-only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	RFRC	TRFC	0	0	0	0	CMDAU	CMDDU	RXT
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR1	RDR0	TDE1	TDE0	ROE1	ROE0	TUE1	TUE0	TFS	RFS	TLS	RLS	RFF1	RFF0	TFE1	TFE0
W																
RESET	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1

Figure 56-28. SSI Interrupt Status Register

Table 56-13. SSI Interrupt Status Register Field Descriptions

Field	Description
31–19	Reserved
24 RFRC	Receive Frame Complete. This flag is set at the end of the frame during which Receiver is disabled. If Receive Frame & Clock are not disabled in the same frame, this flag is also set at the end of the frame in which Receive Frame & Clock are disabled. See description of RFR_CLK_DIS (add cross reference) bit for more details on how to disable Receiver Frame & Clock or keep them enabled after receiver is disabled. 0 End of Frame not reached 1 End of frame reached after disabling RE or disabling RFR_CLK_DIS, when receiver is already disabled.
23 TFRC	Transmit Frame Complete. This flag is set at the end of the frame during which Transmitter is disabled. If Transmit Frame & Clock are not disabled in the same frame, this flag is also set at the end of the frame in which Transmit Frame & Clock are disabled. See description of TFR_CLK_DIS (add cross reference) bit for more details on how to disable Receiver Frame & Clock or keep them enabled after receiver is disabled. 0 End of Frame not reached 1 End of frame reached after disabling TE or disabling TFR_CLK_DIS, when transmitter is already disabled.
18 CMDAU	Command Address Register Updated. This bit causes the Command Address Updated interrupt (when CMDAU_EN bit is set). This status bit is set each time there is a difference in the previous and current value of the received Command Address. This bit is cleared on reading the SACADD register. 0 No change in SACADD register. 1 SACADD register updated with different value.
17 CMDDU	Command Data Register Updated. This bit causes the Command Data Updated interrupt (when CMDDU_EN bit is set). This status bit is set each time there is a difference in the previous and current value of the received Command Data. This bit is cleared on reading the SACDAT register. 0 No change in SACDAT register. 1 SACDAT register updated with different value.

Table 56-13. SSI Interrupt Status Register Field Descriptions (continued)

Field	Description
16 RXT	Receive Tag Updated. This status bit is set each time there is a difference in the previous and current value of the received tag. It causes the Receive Tag Interrupt (if RXT_EN bit is set). This bit is cleared on reading the SATAG register. 0 No change in SATAG register. 1 SATAG register updated with different value.
15 RDR1	Receive Data Ready 1. This flag bit is set when SRX1 or Rx FIFO 1 is loaded with a new value and Two-Channel mode is selected. RDR1 is cleared when the Core reads the SRX1 register. If Rx FIFO 1 is enabled, RDR1 is cleared when the FIFO is empty. If RIE and RDR1_EN are set, a Receive Data 1 interrupt request is issued on setting of RDR1 bit in case Rx FIFO1 is disabled, if the FIFO is enabled, the interrupt is issued on RFF1 assertion. The RDR1 bit is cleared by POR and SSI reset. 0 No new data for Core to read. 1 New data for Core to read.
14 RDR0	Receive Data Ready 0. This flag bit is set when SRX0 or Rx FIFO 0 is loaded with a new value. RDR0 is cleared when the Core reads the SRX0 register. If Rx FIFO 0 is enabled, RDR0 is cleared when the FIFO is empty. If RIE and RDR0_EN are set, a Receive Data 0 interrupt request is issued on setting of RDR0 bit in case Rx FIFO0 is disabled, if the FIFO is enabled, the interrupt is issued on RFF0 assertion. The RDR0 bit is cleared by POR and SSI reset. 0 No new data for Core to read. 1 New data for Core to read.
13 TDE1	Transmit Data Register Empty 1. This flag is set whenever data is transferred to TXSR from STX1 register and Two-Channel mode is selected. If Tx FIFO1 is enabled, this occurs when there is at least one empty slot in STX1 or Tx FIFO1. If Tx FIFO1 is not enabled, this occurs when the contents of STX1 are transferred to TXSR. The TDE1 bit is cleared when the Core writes to STX1. If TIE and TDE1_EN are set, an SSI Transmit Data 1 interrupt request is issued on setting of TDE1 bit. The TDE1 bit is cleared by POR and SSI reset. 0 Data available for transmission. 1 Data needs to be written by the Core for transmission.
12 TDE0	Transmit Data Register Empty 0. This flag is set whenever data is transferred to TXSR from STX0 register. If Tx FIFO 0 is enabled, this occurs when there is at least one empty slot in STX0 or Tx FIFO 0. If Tx FIFO 0 is not enabled, this occurs when the contents of STX0 are transferred to TXSR. The TDE0 bit is cleared when the Core writes to STX0. If TIE and TDE0_EN are set, an SSI Transmit Data 0 interrupt request is issued on setting of TDE0 bit. The TDE0 bit is cleared by POR and SSI reset. 0 Data available for transmission. 1 Data needs to be written by the Core for transmission.
11 ROE1	Receiver Overrun Error 1. This flag is set when the RXSR is filled and ready to transfer to SRX1 register or to Rx FIFO 1 (when enabled) and these are already full and Two-Channel mode is selected. If Rx FIFO 1 is enabled, this is indicated by RFF1 flag, else this is indicated by the RDR1 flag. The RXSR is not transferred in this case. The ROE1 flag causes an interrupt if RIE and ROE1_EN are set. The ROE1 bit is cleared by POR and SSI reset. It is also cleared by writing ‘1’ to this bit. Clearing the RE bit does not affect the ROE1 bit. 0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.

Table 56-13. SSI Interrupt Status Register Field Descriptions (continued)

Field	Description
10 ROE0	Receiver Overrun Error 0. This flag is set when the RXSR is filled and ready to transfer to SRX0 register or to Rx FIFO 0 (when enabled) and these are already full. If Rx FIFO 0 is enabled, this is indicated by RFF0 flag, else this is indicated by the RDR0 flag. The RXSR is not transferred in this case. The ROE0 flag causes an interrupt if RIE and ROE0_EN are set. The ROE0 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit. Clearing the RE bit does not affect the ROE0 bit. 0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.
9 TUE1	Transmitter Underrun Error 1. This flag is set when the TXSR is empty (no data to be transmitted), the TDE1 flag is set, a transmit time slot occurs and the SSI is in Two-Channel mode. When a transmit underrun error occurs, the previous data is retransmitted. In Network mode, each time slot requires data transmission (unless masked through STMSK register), when the transmitter is enabled (TE is set). The TUE1 flag causes an interrupt if TIE and TUE1_EN are set. The TUE1 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit. 0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.
8 TUE0	Transmitter Underrun Error 0. This flag is set when the TXSR is empty (no data to be transmitted), the TDE0 flag is set and a transmit time slot occurs. When a transmit underrun error occurs, the previous data is retransmitted. In Network mode, each time slot requires data transmission (unless masked through STMSK register), when the transmitter is enabled (TE is set). The TUE0 flag causes an interrupt if TIE and TUE0_EN are set. The TUE0 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit. 0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.
7 TFS	Transmit Frame Sync. This flag indicates the occurrence of transmit frame sync. Data written to the STX registers during the time slot when the TFS flag is set, is sent during the second time slot (in Network mode) or in the next first time slot (in Normal mode). In Network mode, the TFS bit is set during transmission of the first time slot of the frame and is then cleared when starting transmission of the next time slot. In Normal mode, this bit is high for the first time slot. This flag causes an interrupt if TIE and TFS_EN are set. The TFS bit is cleared by POR and SSI reset. 0 No Occurrence of Transmit frame sync. 1 Transmit frame sync occurred during transmission of last word written to STX registers.
6 RFS	Receive Frame Sync. This flag indicates the occurrence of receive frame sync. In Network mode, the RFS bit is set when the first slot of the frame is being received. It is cleared when the next slot begins to be received. In Normal mode, this bit is always high. This flag causes an interrupt if RIE and RFS_EN are set. The RFS bit is cleared by POR and SSI reset. 0 No Occurrence of Receive frame sync. 1 Receive frame sync occurred during reception of next word in SRX registers.
5 TLS	Transmit Last Time Slot. This flag indicates the last time slot in a frame. When set, it indicates that the current time slot is the last time slot of the frame. TLS is set at the start of the last transmit time slot and causes the SSI to issue an interrupt (if TIE and TLS_EN are set). TLS is not generated when frame rate is 1 in normal mode of operation. TLS is cleared when the SISR is read with this bit set. The TLS bit is cleared by POR and SSI reset. 0 Current time slot is not last time slot of frame. 1 Current time slot is the last transmit time slot of frame.

Table 56-13. SSI Interrupt Status Register Field Descriptions (continued)

Field	Description
4 RLS	Receive Last Time Slot. This flag indicates the last time slot in a frame. When set, it indicates that the current time slot is the last receive time slot of the frame. RLS is set at the end of the last time slot and causes the SSI to issue an interrupt (if RIE and RLS_EN are set). RLS is cleared when the SISR is read with this bit set. The RLS bit is cleared by POR and SSI reset. 0 Current time slot is not last time slot of frame. 1 Current time slot is the last receive time slot of frame.
3 RFF1	Receive FIFO Full 1. This flag is set when Rx FIFO1 is enabled, the data level in Rx FIFO1 reaches the selected Rx FIFO WaterMark 1 (RFWM1) threshold and the SSI is in Two-Channel mode. The setting of RFF1 only causes an interrupt when RIE and RFF1_EN are set, Rx FIFO1 is enabled and the Two-Channel mode is selected. RFF1 is automatically cleared when the amount of data in Rx FIFO1 falls below the threshold. The RFF1 bit is cleared by POR and SSI reset. When Rx FIFO1 contains 15 words, the maximum it can hold, all further data received (for storage in this FIFO) is ignored until the FIFO contents are read. 0 Space available in Receive FIFO1. 1 Receive FIFO1 is full.
2 RFF0	Receive FIFO Full 0. This flag is set when Rx FIFO0 is enabled and the data level in Rx FIFO0 reaches the selected Rx FIFO WaterMark 0 (RFWM0) threshold. The setting of RFF0 only causes an interrupt when RIE and RFF0_EN are set and Rx FIFO0 is enabled. RFF0 is automatically cleared when the amount of data in Rx FIFO0 falls below the threshold. The RFF0 bit is cleared by POR and SSI reset. When Rx FIFO0 contains 15 words, the maximum it can hold, all further data received (for storage in this FIFO) is ignored until the FIFO contents are read. 0 Space available in Receive FIFO0. 1 Receive FIFO0 is full.
1 TFE1	Transmit FIFO Empty 1. This flag is set when the empty slots in Tx FIFO exceed or are equal to the selected Tx FIFO WaterMark 1 (TFWM1) threshold and the Two-Channel mode is selected. The setting of TFE1 only causes an interrupt when TIE and TFE1_EN are set, Tx FIFO1 is enabled and Two-Channel mode is selected. The TFE1 bit is automatically cleared when the data level in Tx FIFO1 becomes more than the amount specified by the watermark bits. The TFE1 bit is set by POR and SSI reset. 0 Transmit FIFO1 has data for transmission. 1 Transmit FIFO1 is empty.
0 TFE0	Transmit FIFO Empty 0. This flag is set when the empty slots in Tx FIFO exceed or are equal to the selected Tx FIFO WaterMark 0 (TFWM0) threshold. The setting of TFE0 only causes an interrupt when TIE and TFE0_EN are set and Tx FIFO0 is enabled. The TFE0 bit is automatically cleared when the data level in Tx FIFO0 becomes more than the amount specified by the watermark bits. The TFE0 bit is set by POR and SSI reset. 0 Transmit FIFO0 has data for transmission. 1 Transmit FIFO0 is empty.

56.3.3.9 SSI Interrupt Enable Register (SIER)

The SSI Interrupt Enable Register (SIER) is a 25-bit register used to set up the SSI interrupts and DMA requests. See Figure 56-29 for illustration of valid bits in SSI Interrupt Enable Register and Table 56-14 for description of the bit fields in the register.

0x43FA+0x0018 (SIER) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	RFRC_EN	TFRC_EN	RDMAE	RIE	TDMAE	TIE	CMDAU_EN	CMDDU_EN	RXT_EN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR1_EN	RDR0_EN	TDE1_EN	TDE0_EN	ROE1_EN	ROE0_EN	TUE1_EN	TUE0_EN	TFS_EN	RFS_EN	TLS_EN	RLS_EN	RFF1_EN	RFF0_EN	TFE1_EN	TFE0_EN
W																
RESET	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1

Figure 56-29. SSI Interrupt Enable Register

Table 56-14. SSI Interrupt Enable Register Field Descriptions

Field	Description
31–23	Reserved
24-23 Enable Bits	Enable Bit. Each bit controls whether the corresponding status bit in SISR can issue an interrupt to the Core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
22 RDMAE	Receive DMA Enable. This bit allows SSI to request for DMA transfers. When enabled, DMA requests are generated when any of the RFF0/1 bits in the SISR are set and if the corresponding RFEN bit is also set. If the corresponding FIFO is disabled, a DMA request is generated when the corresponding RDR bit is set. 0 SSI Receiver DMA requests disabled. 1 SSI Receiver DMA requests enabled.
21 RIE	Receive Interrupt Enable. This control bit allows the SSI to issue receiver related interrupts to the Core. Refer to Section 56.4.3 for a detailed description of this bit. 0 SSI Receiver Interrupt requests disabled. 1 SSI Receiver Interrupt requests enabled.
20 TDMAE	Transmit DMA Enable. This bit allows SSI to request for DMA transfers. When enabled, DMA requests are generated when any of the TFE0/1 bits in the SISR are set and if the corresponding TFEN bit is also set. If the corresponding FIFO is disabled, a DMA request is generated when the corresponding TDE bit is set. 0 SSI Transmitter DMA requests disabled. 1 SSI Transmitter DMA requests enabled.

Table 56-14. SSI Interrupt Enable Register Field Descriptions (continued)

Field	Description
19 TIE	Transmit Interrupt Enable. This control bit allows the SSI to issue transmitter data related interrupts to the Core. Refer to Section 56.4.4 for a detailed description of this bit. 0 SSI Transmitter Interrupt requests disabled. 1 SSI Transmitter Interrupt requests enabled.
18–0 Enable Bits	Enable Bit. Each bit controls whether the corresponding status bit in SISR can issue an interrupt to the Core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.

56.3.3.10 SSI Transmit Configuration Register (STCR)

The SSI Transmit Configuration Register (STCR) is a read/write control registers used to direct the transmit operation of the SSI. STCR controls the direction of the bit clock and frame sync ports, STCK and STFS. Interrupt enable bit for the transmit sections is provided in this control register. The Power-on reset clears all STCR bits. However, SSI reset does not affect the STCR bits. The STCR bits are described in the following paragraphs. See Table 56-7 for the programming model of the SSI. The SSI Control Register (SCR) must first be set to enable interrupts. Next, the SSI interrupt bit in the Interrupt Enable Register (SIER) must be set to enable the interrupt. Finally, the interrupt can be enabled from within the SSI.

See Figure 56-30 for illustration of valid bits in SSI Transmit Configuration Register and Table 56-15 for description of the bit fields in the register.

0x43FA+0x001C (STCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	TXBIT0	TFEN1	TFEN0	TFDIR	TXDIR	TSHFD	TSCKP	TFSI	TFSL	TEFS
W																
RESET	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Figure 56-30. SSI Transmit Configuration Register

Table 56-15. SSI Transmit Configuration Register Field Descriptions

Field	Description
31–10	Reserved
9 TXBIT0	Transmit Bit 0. This control bit allows SSI to transmit the data word from bit position 0 or 15/31 in the transmit shift register. The shifting data direction can be MSB or LSB first, controlled by the TSHFD bit. 0 Shifting with respect to bit 31 (if word length = 16, 18, 20, 22 or 24) or bit 15 (if word length = 8, 10 or 12) of transmit shift register (MSB aligned). 1 Shifting with respect to bit 0 of transmit shift register (LSB aligned).
8 TFEN1	Transmit FIFO Enable 1. This bit enables transmit FIFO 1. When enabled, the FIFO allows 15 samples to be transmitted by the SSI (per channel) (a 9th sample can be shifting out) before TDE1 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is transferred to the transmit shift register (provided the interrupt is enabled). 0 Transmit FIFO 1 disabled. 1 Transmit FIFO 1 enabled.
7 TFEN0	Transmit FIFO Enable 0. This bit enables transmit FIFO 0. When enabled, the FIFO allows 15 samples to be transmitted by the SSI per channel (a 9th sample can be shifting out) before TDE0 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is transferred to the transmit shift register (provided the interrupt is enabled). 0 Transmit FIFO 0 disabled. 1 Transmit FIFO 0 enabled.
6 TFDIR	Transmit Frame Direction. This bit controls the direction and source of the transmit frame sync signal. Internally generated frame sync signal is sent out through the STFS port and external frame sync is taken from the same port. 0 Frame Sync is external. 1 Frame Sync generated internally.
5 TXDIR	Transmit Clock Direction. This bit controls the direction and source of the clock signal used to clock the TXSR. Internally generated clock is output through the STCK port. External clock is taken from this port. Refer to Table 56-5 for details of clock pin configurations 0 Transmit Clock is external. 1 Transmit Clock generated internally.
4 TSHFD	Transmit Shift Direction. This bit controls whether the MSB or LSB will be transmitted first in a sample. **Note:** The CODEC device labels the MSB as bit 0, whereas the Core labels the LSB as bit 0. Therefore, when using a standard CODEC, Core MSB (CODEC LSB) is shifted in first (TSHFD cleared). 0 Data transmitted MSB first. 1 Data transmitted LSB first.
3 TSCKP	Transmit Clock Polarity. This bit controls which bit clock edge is used to clock out data for the transmit section. 0 Data clocked out on rising edge of bit clock. 1 Data clocked out on falling edge of bit clock. Note: TSCKP is 0 CLK_IST = 1; TSCKP is 1CLK_IST = 0
2 TFSI	Transmit Frame Sync Invert. This bit controls the active state of the frame sync I/O signal for the transmit section of SSI. 0 Transmit frame sync is active high. 1 Transmit frame sync is active low.

Table 56-15. SSI Transmit Configuration Register Field Descriptions (continued)

Field	Description
1 TFSL	Transmit Frame Sync Length. This bit controls the length of the frame sync signal to be generated or recognized for the transmit section. The length of a word-long frame sync is same as the length of the data word selected by WL[3:0]. 0 Transmit frame sync is one-word long. 1 Transmit frame sync is one-clock-bit long.
0 TEFS	Transmit Early Frame Sync. This bit controls when the frame sync is initiated for the transmit section. The frame sync signal is deasserted after one bit-for-bit length frame sync and after one word-for-word length frame sync. In case of synchronous operation, the frame sync can also be initiated on receiving the first bit of data. 0 Transmit frame sync initiated as the first bit of data is transmitted. 1 Transmit frame sync is initiated one bit before the data is transmitted.

56.3.3.11 SSI Receive Configuration Register (SRCR)

The SSI Receive Configuration Register (SRCR) is a read/write control registers used to direct the receive operation of the SSI. SRCR controls the direction of the bit clock and frame sync ports, SRCK and SRFS. Interrupt enable bit for the transmit sections is provided in this control register. The Power-on reset clears all SRCR bits. However, SSI reset does not affect the SRCR bits. See Figure 56-31 for illustration of valid bits in SSI Receive Configuration Register and Table 56-16 for description of the bit fields in the register.

0x43FA+0x0020 (SRCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	RXEXT	RXBIT0	RFEN1	RFEN0	RFDIR	RXDIR	RSHFD	RSCKP	RFSI	RFSL	REFS
W																
RESET	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Figure 56-31. SSI Receive Configuration Register

Table 56-16. SSI Receive Configuration Register Field Descriptions

Field	Description
31–11	Reserved
10 RXEXT	Receive Data Extension. This control bit allows SSI to store the received data word in sign extended form. This bit affects data storage only in case received data is LSB aligned (SRCR[9]=1) 0 Sign extension turned off. 1 Sign extension turned on.

Table 56-16. SSI Receive Configuration Register Field Descriptions (continued)

Field	Description
9 RXBIT0	Receive Bit 0. This control bit allows SSI to receive the data word at bit position 0 or 15/31 in the receive shift register. The shifting data direction can be MSB or LSB first, controlled by the RSHFD bit. 0 Shifting with respect to bit 31 (if word length = 16, 18, 20, 22 or 24) or bit 15 (if word length = 8, 10 or 12) of receive shift register (MSB aligned). 1 Shifting with respect to bit 0 of receive shift register (LSB aligned).
8 RFEN1	Receive FIFO Enable 1. This bit enables receive FIFO 1. When enabled, the FIFO allows 15 samples to be received by the SSI per channel (a 16th sample can be shifting in) before RDR1 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is received by the SSI (provided the interrupt is enabled). 0 Receive FIFO 1 disabled. 1 Receive FIFO 1 enabled.
7 RFEN0	Receive FIFO Enable 0. This bit enables receive FIFO 0. When enabled, the FIFO allows 15 samples to be received by the SSI (per channel) (a 16th sample can be shifting in) before RDR0 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is received by the SSI (provided the interrupt is enabled). 0 Receive FIFO 0 disabled. 1 Receive FIFO 0 enabled.
6 RFDIR	Receive Frame Direction. This bit controls the direction and source of the receive frame sync signal. Internally generated frame sync signal is sent out through the SRFS port and external frame sync is taken from the same port. 0 Frame Sync is external. 1 Frame Sync generated internally.
5 RXDIR	Receive Clock Direction. This bit controls the direction and source of the clock signal used to clock the RXSR. Internally generated clock is output through the SRCK port. External clock is taken from this port. Refer to Table 56-5 for details on clock pin configurations. 0 Receive Clock is external. 1 Receive Clock generated internally.
4 RSHFD	Receive Shift Direction. This bit controls whether the MSB or LSB will be received first in a sample. **Note:** The CODEC device labels the MSB as bit 0, whereas the Core labels the LSB as bit 0. Therefore, when using a standard CODEC, Core MSB (CODEC LSB) is shifted in first (RSHFD cleared). 0 Data received MSB first. 1 Data received LSB first.
3 RSCKP	Receive Clock Polarity. This bit controls which bit clock edge is used to latch in data for the receive section. 0 Data latched on rising edge of bit clock. 1 Data latched on falling edge of bit clock.
2 RFSI	Receive Frame Sync Invert. This bit controls the active state of the frame sync I/O signal for the receive section of SSI. 0 Receive frame sync is active high. 1 Receive frame sync is active low.

Table 56-16. SSI Receive Configuration Register Field Descriptions (continued)

Field	Description
1 RFSL	Receive Frame Sync Length. This bit controls the length of the frame sync signal to be generated or recognized for the receive section. The length of a word-long frame sync is same as the length of the data word selected by WL[3:0]. 0 Receive frame sync is one-word long. 1 Receive frame sync is one-clock-bit long.
0 REFS	Receive Early Frame Sync. This bit controls when the frame sync is initiated for the receive section. The frame sync is disabled after one bit-for-bit length frame sync and after one word-for-word length frame sync. 0 Receive frame sync initiated as the first bit of data is received. 1 Receive frame sync is initiated one bit before the data is received.

56.3.3.12 SSI Transmit and Receive Clock Control Registers (STCCR & SRCCR)

The SSI Transmit and Receive Control (STCCR and SRCCR) registers are 19-bit, read/write control registers used to direct the operation of the SSI. The Clock and Reset Module (CRM) can source the SSI clock (ccm_ssi_clk) from multiple sources and perform fractional division to support commonly used audio bit rates. The CRM can maintain the ccm_ssi_clk frequency at a constant rate even in cases where the ipg_clk frequency changes. These registers control the SSI clock generator, bit and frame sync rates, word length, and number of words per frame for the serial data. The STCCR register is dedicated to the transmit section, and the SRCCR register is dedicated to the receive section except in Synchronous mode, in which the STCCR register controls both the receive and transmit sections. Power-on reset clears all STCCR and SRCCR bits. SSI reset does not affect the STCCR and SRCCR bits. The control bits are described in the following paragraphs. Although the bit patterns of the STCCR and SRCCR registers are the same, the contents of these two registers can be programmed differently.

See Figure 56-32 for illustration of valid bits in SSI Transmit Clock Control Register and Table 56-17 for description of the bit fields for both SSI Transmit and Receive Clock Control registers.

0x43FA+0x0024 (STCCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DIV2	PSR	WL3
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WL2	WL1	WL0	DC4	DC3	DC2	DC1	DC0	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-32. SSI Transmit Clock Control Register

See Figure 56-33 for illustration of valid bits in SSI Receive Clock Control Register and Table 56-17 for description of the bit fields for both SSI Transmit and Receive Clock Control registers.

0x43FA+0x0028 (SRCCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DIV2	PSR	WL3
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WL2	WL1	WL0	DC4	DC3	DC2	DC1	DC0	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-33. SSI Receive Clock Control Register

Table 56-17. SSI Transmit and Receive Clock Control Register Field Descriptions

Field	Description
31–19	Reserved
18 DIV2	Divide By 2. This bit controls a divide-by-two divider in series with the rest of the prescalers. 0 Divider bypassed. 1 Divider used to divide clock by 2.
17 PSR	Prescaler Range. This bit controls a fixed divide-by-eight prescaler in series with the variable prescaler. It extends the range of the prescaler for those cases where a slower bit clock is required. 0 Prescaler bypassed. 1 Prescaler used to divide clock by 8.
16–13 WL3–WL0	Word Length Control. These bits are used to control the length of the data words being transferred by the SSI. These bits control the Word Length Divider in the Clock Generator. They also control the frame sync pulse length when the FSL bit is cleared. In I2S Master mode, the SSI works with a fixed word length of 32, and the WL bits are used to control the amount of valid data in those 32 bits. Refer to Table 56-18 for details about the data word lengths supported by SSI. In AC97 Mode of operation, if word length is set to any value other than 16 bits, it will result in a word length of 20 bits.
12–8 DC4–DC0	Frame Rate Divider Control. These bits are used to control the divide ratio for the programmable frame rate dividers. The divide ratio works on the word clock. In Normal mode, this ratio determines the word transfer rate. In Network mode, this ratio sets the number of words per frame. The divide ratio ranges from 1 to 32 in Normal mode and from 2 to 32 in Network mode. In Normal mode, a divide ratio of 1 (DC=00000) provides continuous periodic data word transfer. A bit-length frame sync must be used in this case. These bits can be programmed with values ranging from “00000” to “11111” to control the number of words in a frame.
7–0 PM7–PM0	Prescaler Modulus Select. These bits control the prescale divider in the clock generator. This prescaler is used only in Internal Clock mode to divide the internal clock (ccm_ssi_clk). The bit clock output is available at the clock port. A divide ratio from 1 to 256 (PM[7:0] = 0x00 to 0xFF) can be selected. Refer to Section 56.4.2.2 for details regarding settings.

Table 56-18. SSI Data Length

WL3	WL2	WL1	WL0	Number of Bits/Word	Supported in Implementation
0	0	0	0	2	No
0	0	0	1	4	No
0	0	1	0	6	No
0	0	1	1	8	Yes
0	1	0	0	10	Yes
0	1	0	1	12	Yes
0	1	1	0	14	No
0	1	1	1	16	Yes
1	0	0	0	18	Yes
1	0	0	1	20	Yes
1	0	1	0	22	Yes
1	0	1	1	24	Yes
1	1	0	0	26	No
1	1	0	1	28	No
1	1	1	0	30	No
1	1	1	1	32	No

56.3.3.13 SSI FIFO Control/Status Register (SFCSR)

See Figure 56-34 for illustration of valid bits in SSI FIFO Control/Status Register and Table 56-20 for description of the bit fields in the register.

0x43FA+0x002C (SFCSR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	RFCNT1[3:0]				TFCNT1[3:0]				RFWM1[3:0]				TFWM1[3:0]			
RESET	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	RFCNT0[3:0]				TFCNT0[3:0]				RFWM0[3:0]				TFWM0[3:0]			
RESET	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Figure 56-34. SSI FIFO Control/Status Register

Table 56-20. SSI FIFO Control/Status Register Field Descriptions

Field	Description
31–28 RFCNT1[3–0]	Receive FIFO Counter 1 These bits indicate the number of data words in Receive FIFO 1. The settings for receive FIFO counter bits are as follows: 0000 0 data word in receive FIFO 0001 1 data word in receive FIFO 0010 2 data word in receive FIFO 0011 3 data word in receive FIFO 0100 4 data word in receive FIFO 0101 5 data word in receive FIFO 0110 6 data word in receive FIFO 0111 7 data word in receive FIFO 1000 8 data word in receive FIFO 1001 9 data word in receive FIFO 1010 10 data word in receive FIFO 1011 11 data word in receive FIFO 1100 12 data word in receive FIFO 1101 13 data word in receive FIFO 1110 14 data word in receive FIFO 1111 15 data word in receive FIFO
27–24 TFCNT1[3–0]	Transmit FIFO Counter 1 These bits indicate the number of data words in Transmit FIFO. The details regarding the settings for transmit FIFO counter bits are as follows: 0000 0 data word in transmit FIFO 0001 1 data word in transmit FIFO 0010 2 data word in transmit FIFO 0011 3 data word in transmit FIFO 0100 4 data word in transmit FIFO 0101 5 data word in transmit FIFO 0110 6 data word in transmit FIFO 0111 7 data word in transmit FIFO 1000 8 data word in transmit FIFO 1001 9 data word in transmit FIFO 1010 10 data word in transmit FIFO 1011 11 data word in transmit FIFO 1100 12 data word in transmit FIFO 1101 13 data word in transmit FIFO 1110 14 data word in transmit FIFO 1111 15 data word in transmit FIFO

Table 56-20. SSI FIFO Control/Status Register Field Descriptions (continued)

Field	Description
23–20 RFWM1[3–0]	Receive FIFO Full WaterMark 1. These bits control the threshold at which the RFF1 flag will be set. The RFF1 flag is set whenever the data level in Rx FIFO 1 reaches the selected threshold. The details regarding the settings for receive FIFO watermark bits are as follows: 0000 Reserved 0001 RFF set when at least one data word have been written to the Receive FIFO Set when RxFIFO = 1,2.....15 data words 0010 RFF set when more than or equal to 2 data word have been written to the Receive FIFO. Set when RxFIFO = 2,3.....15 data words 0011 RFF set when more than or equal to 3 data word have been written to the Receive FIFO. Set when RxFIFO = 3,4.....15 data words 0100 RFF set when more than or equal to 4 data word have been written to the Receive FIFO. Set when RxFIFO = 4,5.....15 data words 0101 RFF set when more than or equal to 5 data word have been written to the Receive FIFO. Set when RxFIFO = 5,6.....15 data words 0110 RFF set when more than or equal to 6 data word have been written to the Receive. Set when RxFIFO = 6,7.....15 data words 0111 RFF set when more than or equal to 7 data word have been written to the Receive FIFO. Set when RxFIFO = 7,8.....15 data words 1000 RFF set when more than or equal to 8 data word have been written to the Receive FIFO. Set when RxFIFO = 8,9.....15 data words 1001 RFF set when more than or equal to 9 data word have been written to the Receive FIFO. Set when RxFIFO = 9,10.....15 data words 1010 RFF set when more than or equal to 10 data word have been written to the Receive FIFO. Set when RxFIFO = 10,11.....15 data words 1011 RFF set when more than or equal to 11 data word have been written to the Receive FIFO. Set when RxFIFO = 11,12.....15 data words 1100 RFF set when more than or equal to 12 data word have been written to the Receive FIFO. Set when RxFIFO = 12,13.....15 data words 1101 RFF set when more than or equal to 13 data word have been written to the Receive FIFO. Set when RxFIFO = 13,14,15data words 1110 RFF set when more than or equal to 14 data word have been written to the Receive FIFO. Set when RxFIFO = 14,15 data words 1111 RFF set when 15 data word have been written to the Receive FIFO (default). Set when RxFIFO = 15 data words

Table 56-20. SSI FIFO Control/Status Register Field Descriptions (continued)

Field	Description
19–16 TFWM1[3–0]	Transmit FIFO Empty WaterMark 1. These bits control the threshold at which the TFE1 flag will be set. The TFE1 flag is set whenever the empty slots in Tx FIFO exceed or are equal to the selected threshold. The details regarding the settings for transmit FIFO watermark bits are as follows: 0000 Reserved 0001 TFE set when there are more than or equal to 1 empty slots in Transmit FIFO. (default) Transmit FIFO empty is set when TxFIFO ≤ 14 data. 0010 TFE set when there are more than or equal to 2 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 13 data. 0011 TFE set when there are more than or equal to 3 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 12 data. 0100 TFE set when there are more than or equal to 4 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 11 data. 0101 TFE set when there are more than or equal to 5 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 10 data. 0110 TFE set when there are more than or equal to 6 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 9 data. 0111 TFE set when there are more than or equal to 7 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 8 data. 1000 TFE set when there are more than or equal to 8 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 7 data. 1001 TFE set when there are more than or equal to 9 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 6 data. 1010 TFE set when there are more than or equal to 10 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 5 data. 1011 TFE set when there are more than or equal to 11 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 4 data. 1100 TFE set when there are more than or equal to 12 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 3 data. 1101 TFE set when there are more than or equal to 13 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 2 data. 1110 TFE set when there are more than or equal to 14 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 1 data. 1111 TFE set when there are 15 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO = 0 data.
15–12 RFCNT0[3–0]	Receive FIFO Counter 0 These bits indicate the number of data words in Receive FIFO 0. Refer to the Receive FIFO Counter 1 bit field description for details regarding the settings for the receive FIFO counter bits.
11–8 TFCNT0[3–0]	Transmit FIFO Counter 0 These bits indicate the number of data words in Transmit FIFO 0. Refer to the Transmit FIFO Counter 1 bit field description for details regarding settings for transmit FIFO counter bits.
7–4 RFWM0[3–0]	Receive FIFO Full WaterMark 0 These bits control the threshold at which the RFF0 flag will be set. The RFF0 flag is set whenever the data level in Rx FIFO 0 reaches the selected threshold. Refer to the Receive FIFO full WaterMark 1 bit field description for details regarding settings for receive FIFO watermark bits.
3–0 TFWM0[3–0]	Transmit FIFO Empty WaterMark 0 These bits control the threshold at which the TFE0 flag will be set. The TFE0 flag is set whenever the empty slots in Tx FIFO exceed or are equal to the selected threshold. Refer to the Transmit FIFO Empty WaterMark 1 bit field description for details regarding settings for transmit FIFO watermark bits.

Table 56-19 indicates the status of the Transmit FIFO Empty flag, with different settings of the Transmit FIFO WaterMark bits and varying amounts of data in the Tx FIFO.

Table 56-19. Status of Transmit FIFO Empty Flag

Transmit FIFO Watermark (TFWM)	Number of data in Tx-Fifo														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
2	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
3	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
7	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
8	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
9	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
10	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
12	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
13	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

56.3.3.14 SSI Test Register (STR)

See Figure 56-21 for illustration of valid bits in SSI Test Register and Table 56-20 for description of the bit fields for the register.

NOTE

SSI Test Register is designed for debugging purpose only and is not intended for general user access.

0x43FA+0x0030 (STR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TEST	RCK2 TCK	RFS2 TFS	RXSTATE[4:0]					TXD2 RXD	TCK2 RCK	TFS2 RFS	TXSTATE[4:0]				
RESET	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1

Figure 56-21. SSI Test Register

Table 56-20. SSI Test Register Field Descriptions

Field	Description
31–16	Reserved
15 TEST	Test Mode. This bit enables the test features of SSI. When set, the RXSTATE and TXSTATE bit values get transferred to the state machine. When in test mode, the user can read the contents of a specific FIFO by writing an appropriate value to the RFCNT0/1 or TFCNT 0/1 bits (in SFCSR). 0 No effect on SSI operation. 1 SSI in Test Mode.
14 RCK2TCK	Receive Clock to Transmit Clock Loop Back. This bit connects SRCK (used as output) to STCK (used as input). 0 No effect on SSI operation. 1 SRCK to STCK loop back enabled.
13 RFS2TFS	Receive Frame to Transmit Frame Loop Back. This bit connects SRFS (used as output) to STFS (used as input). 0 No effect on SSI operation. 1 SRFS to STFS loop back enabled.
12–8 RXSTATE[4:0]	Receiver State Machine Status. These bits indicate the current status of the Receive State Machine.
7 TXD2RXD	Transmit Data to Receive Data Loop Back. This bit connects STXD to SRXD. 0 No effect on SSI operation. 1 STXD to SRXD loop back enabled.

Table 56-20. SSI Test Register Field Descriptions (continued)

Field	Description
6 TCK2RCK	Transmit Clock to Receive Clock Loop Back. This bit connects STCK (used as output) to SRCK (used as input). 0 No effect on SSI operation. 1 STCK to SRCK loop back enabled.
5 TFS2RFS	Transmit Frame to Receive Frame Loop Back. This bit connects STFS (used as output) to SRFS (used as input). 0 No effect on SSI operation. 1 STFS to SRFS loop back enabled.
4–0 TXSTATE	Transmitter State Machine Status. These bits indicate the current status of the Transmit State Machine.

56.3.3.15 SSI Option Register (SOR)

See Figure 56-22 for illustration of valid bits in SSI Option Register and Table 56-21 for description of the bit fields for the register.

NOTE

SSI Option Register is designed for debugging purpose only and is not intended for general user access.

0x43FA+0x0034 (SOR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	CLKOFF	RX_CLR	TX_CLR	INIT	WAIT[1:0]		SYNRST
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-22. SSI Option Register

Table 56-21. SSI Option Register Field Descriptions

Field	Description
31–7	Reserved
6 CLKOFF	Clock Off. This bit is used to turn off the ipg_clk to further reduce power consumption. 0 No effect on SSI operation. 1 Turn off ipg_clk.

Table 56-21. SSI Option Register Field Descriptions (continued)

Field	Description
5 RX_CLR	Receiver Clear. This bit flushes the contents of Rx FIFOs. It is always read as cleared (‘0’). 0 No effect on SSI operation. 1 Flush Rx FIFOs.
4 TX_CLR	Transmitter Clear This bit flushes the contents of Tx FIFOs. It is always read as cleared (‘0’). 0 No effect on SSI operation. 1 Flush Tx FIFOs. **It is recommended that we flush Tx-FIFOs after re-enabling Transmission.
3 INIT	Initialize. The setting of this bit causes the SSI state machine to reset. 0 No effect on SSI operation. 1 Initialize SSI state machine.
2–1 WAIT[1:0]	Wait. These bits control the number wait states to be added to all transactions between the Core and SSI. The value of these bits ranges from ‘00’ (no wait states) to ‘11’ (three wait states).
0 SYNRST	Frame Sync Reset. This bit automatically resets the accumulation of data in Receive Data Registers (SRX0/1) and Receive FIFOs (RXFIFO 0/1) on the next frame synchronization. 0 Data accumulation not affected. 1 Reset data accumulation on Frame Synchronization.

56.3.3.16 SSI AC97 Control Register (SACNT)

See Figure 56-23 for illustration of valid bits in SSI AC97 Control Register and Table 56-22 for description of the bit fields for the register.

0x43FA+0x0038 (SACNT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	FRDIV[5:0]						WR	RD	TIF	FV	AC97EN
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-23. SSI AC97 Control Register

Table 56-22. SSI AC97 Control Register Field Descriptions

Field	Description
31–11	Reserved
10–5 FRDIV[5:0]	Frame Rate Divider. These bits control the frequency of AC97 data transmission/reception. They are programmed with the number of frames for which the SSI should be idle, after operating in one frame. Through these bits, AC97 frequency of operation, from 48 KHz (000000) to 1 KHz (101111) can be achieved. Sample Value: 001010 (10 Decimal) = SSI will operate once every 11 frames.
4 WR	Write Command. This bit specifies whether the next frame will carry an AC97 Write Command or not. The programmer should take care that only one of the bits (WR or RD) is set at a time. When this bit is set, the corresponding tag bits (corresponding to Command Address and Command Data slots of the next Tx frame) are automatically set. This bit is automatically cleared by the SSI after completing transmission of a frame. 0 Next frame will not have a Write Command. 1 Next frame will have a Write Command.
3 RD	Read Command. This bit specifies whether the next frame will carry an AC97 Read Command or not. The programmer should take care that only one of the bits (WR or RD) is set at a time. When this bit is set, the corresponding tag bit (corresponding to Command Address slot of the next Tx frame) is automatically set. This bit is automatically cleared by the SSI after completing transmission of a frame. 0 Next frame will not have a Read Command. 1 Next frame will have a Read Command.
2 TIF	Tag in FIFO. This bit controls the destination of the information received in AC97 tag slot (Slot #0). 0 Tag info stored in SATAG register. 1 Tag info stored in Rx FIFO 0.
1 FV	Fixed/Variable Operation. This bit selects whether the SSI is in AC97 Fixed mode or AC97 Variable mode. 0 AC97 Fixed Mode. 1 AC97 Variable Mode.
0 AC97EN	AC97 Mode Enable. This bit is used to enable SSI AC97 operation. Refer to Section 56.1.2.5 for details of AC97 operation. 0 AC97 mode disabled. 1 SSI in AC97 mode.

56.3.3.17 SSI AC97 Command Address Register (SACADD)

See Figure 56-24 for illustration of valid bits in SSI AC97 Command Address Register and Table 56-23 for description of the bit fields for the register.

0x43FA+0x003C (SACADD) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	SACADD[18:16]		
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SACADD[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-24. SSI AC97 Command Address Register

Table 56-23. SSI AC97 Command Address Register Field Descriptions

Field	Description
31–19	Reserved
18–0 SACADD	AC97 Command Address. These bits store the Command Address Slot information (bit 19 of the slot is sent in accordance with the Read and Write Command bits in SACNT register). These bits can be updated by a direct write from the Core. They are also updated with the information received in the incoming Command Address Slot. If the contents of these bits change due to an update, the CMDAU bit in SISR is set.

56.3.3.18 SSI AC97 Command Data Register (SACDAT)

See Figure 56-25 for illustration of valid bits in SSI AC97 Command Data Register and Table 56-24 for description of the bit fields for the register.

0x43FA+0x0040 (SACDAT) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	SACDAT[19:16]			
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SACDAT[15:0]															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-25. SSI AC97 Command Data Register

Table 56-24. SSI AC97 Command Data Register

Field	Description
31–20	Reserved
19–0 SACDAT	AC97 Command Data. The outgoing Command Data Slot carries the information contained in these bits. These bits can be updated by a direct write from the Core. They are also updated with the information received in the incoming Command Data Slot. If the contents of these bits change due to an update, the CMDDU bit in SISR is set. These bits are transmitted only during AC97 Write Command. During AC97 Read Command, 0x00000 is transmitted in time slot #2.

56.3.3.19 SSI AC97 Tag Register (SATAG)

See Figure 56-26 for illustration of valid bits in SSI AC97 Tag Register and Table 56-25 for description of the bit fields for the register.

0x43FA+0x0044 (SATAG) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SATAG[15:0]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-26. SSI AC97 Tag Register

Table 56-25. SSI AC97 Tag Register Field Descriptions

Field	Description
31–16	Reserved
15–0 SATAG	AC97 Tag Value. Writing to this register (by the Core) sets the value of the Tx-Tag in AC97 fixed mode of operation. On a read, the Core gets the Rx-Tag Value received (in the last frame) from the Codec. If TIF bit in SACNT register is set, the TAG value is also stored in Rx-FIFO in addition to SATAG register. When the received Tag value changes, the RXT bit in SISR register is set. Bits SATAG[1:0] convey the Codec -ID. In current implementation only Primary Codecs are supported. Thus writing value 2'b00 to this field is mandatory.

56.3.3.20 SSI Transmit Time Slot Mask Register (STMSK)

See Figure 56-27 for illustration of valid bits in SSI Transmit Time Slot Register and Table 56-26 for description of the bit fields for the register.

0x43FA+0x0048 (STMSK) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STMSK[31:16]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STMSK[15:0]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-27. SSI Transmit Time Slot Mask Register

Table 56-26. SSI Transmit Time Slot Mask Register Field Descriptions

Field	Description
31–0 STMSK	Transmit Mask. These bits indicate which slot has been masked in the current frame. The Core can write to this register to control the time slots in which the SSI transmits data. Each bit has info corresponding to the respective time slot in the frame. Transmit mask bits should not be used in I2S Slave mode of operation. STMSK register value must be set before enabling transmission. 0 Valid Time Slot. 1 Time Slot masked (no data transmitted in this time slot).

56.3.3.21 SSI Receive Time Slot Mask Register (SRMSK)

See Figure 56-28 for illustration of valid bits in SSI Receive Time Slot Mask Register and Table 56-27 for description of the bit fields for the register.

0x43FA+0x004C (SRMSK) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRMSK[31:16]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRMSK[15:0]															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-28. SSI Receive Time Slot Mask Register

Table 56-27. SSI Receive Time Slot Mask Register Field Descriptions

Field	Description
31–0 SRMSK	Receive Mask. These bits indicate which slot has been masked in the current frame. The core can write to this register to control the time slots in which the SSI receives data. Each bit has info corresponding to the respective time slot in the frame. SRMSK register value must be set before enabing the receiver. Receive mask bits should not be used in I2S Slave mode of operation. 0 Valid Time Slot. 1 Time Slot masked (no data received in this time slot).

56.3.3.22 SSI AC97 Channel Status Register (SACCST)

See Figure 56-29 for illustration of valid bits in SSI AC97 Channel Status Register and Table 56-28 for description of the bit fields for the register.

0x43FA+0x0050 (SACCST) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	SACCST[9:0]									
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-29. SSI AC97 Channel Status Register

Table 56-28. SSI AC97 Channel Status Register Field Descriptions

Field	Description
31–10	Reserved
9–0 SACCST	AC97 Channel Status. These bits indicate which data slot has been enabled in AC97 variable mode operation. This register is updated in case the core enables/disables a channel through a write to SACCEN/SACCDIS register or the external codec enables a channel by sending a ‘1’ in the corresponding SLOTREQ bit. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). The contents of this register only have relevance while the SSI is operating in AC97 variable mode. Writes to this register result in an error response on the IP interface. 0 Data channel disabled. 1 Data channel enabled.

56.3.3.23 SSI AC97 Channel Enable Register (SACCEN)

See Figure 56-30 for illustration of valid bits in SSI AC97 Channel Enable Register and Table 56-29 for description of the bit fields for the register.

0x43FA+0x0054 (SACCEN) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W							SACCEN[9:0]									
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-30. SSI AC97 Channel Enable Register

Table 56-29. SSI AC97 Channel Enable Register Field Descriptions

Field	Description
31–10	Reserved
9–0 SACCEN	AC97 Channel Enable. The Core writes a ‘1’ to these bits to enable an AC97 data channel. Writing a ‘0’ has no effect. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). Writes to these bits only have effect in the AC97 Variable mode of operation. These bits are always read as ‘0’ by the Core. 0 Write Has no effect. 1 Write Enables the corresponding data channel.

56.3.3.24 SSI AC97 Channel Disable Register (SACCDIS)

See Figure 56-31 for illustration of valid bits in SSI AC97 Channel Disable Register and Table 56-30 for description of the bit fields for the register.

0x43FA+0x0058 (SACCDIS) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W							SACCDIS[9:0]									
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 56-31. SSI AC97 Channel Disable Register

Table 56-30. SSI AC97 Channel Disable Register Field Descriptions

Field	Description
31–10	Reserved
9–0 SACCDIS	AC97 Channel Disable. The Core writes a ‘1’ to these bits to disable an AC97 data channel. Writing a ‘0’ has no effect. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). Writes to these bits only have effect in the AC97 Variable mode of operation. These bits are always read as ‘0’ by the Core. 0 Write Has no effect. 1 Write Disables the corresponding data channel.

56.4 Functional Description

This section describes the following:

- Section 56.4.1, SSI Architecture”
- Section 56.4.2, SSI Clocking”
- Section 56.4.3, Receive Interrupt Enable Bit Description”
- Section 56.4.4, Transmit Interrupt Enable Bit Description”
- Section 56.4.5, Internal Frame and Clock Shutdown”
- Section 56.4.6, IP Bus Interface”

56.4.1 SSI Architecture

The Synchronous Serial Interface (SSI) is connected to chip pads through the Digital Audio Mux (AUDMUX) module or directly as well. The AUDMUX can be configured to connect the SSI module to the chip pads in various ways. Refer to Figure 56-1 for a block diagram of the SSI.

56.4.2 SSI Clocking

The SSI uses the following clocks:

- Bit clock—Used to serially clock the data bits in and out of the SSI port. This clock is either generated internally (from ccm_ssi_clk) or taken from external clock source (through the Tx/Rx clock ports).
- Word clock—Used to count the number of data bits per word (8, 10, 12, 16, 18, 20, 22 or 24 bits). This clock is generated internally from the bit clock.
- Frame clock (Frame Sync)—Used to count the number of words in a frame. This signal can be generated internally from the bit clock, or taken from external source (from the Tx/Rx frame sync ports).
- Sys clock—In master mode, this is an integer multiple of frame clock. This is ccm_ssi_clk. It is used in cases when SSI has to provide the clock.

Care should be taken to ensure that the bit clock frequency (either internally generated by dividing the ccm_ssi_clk or sourced from external device through Tx/Rx clock ports) is never greater than 1/5 of the ipg_clk frequency.

In Normal mode (SCR[6:5]=00), the bit clock, used to serially clock the data, is visible on the Serial Transmit Clock (STCK) and Serial Receive Clock (SRCK) ports. The word clock is an internal clock used to determine when transmission of an 8, 10, 12, 16, 18, 20, 22, or 24 bit word has completed. The word clock in turn then clocks the frame clock, which counts the number of words in the frame. The frame clock can be viewed on the STFS and SRFS frame sync ports, because a frame sync is generated after the correct number of words in the frame have passed. In master and synchronous mode, the unused port SRCK is used as Serial Oversampling Clock (ccm_ssi_clk) enabled by the SCR register bit 15, SYS_CLK_EN. This Serial System Clock is an oversampling clock of the frame sync clock (STFS). In this mode, the word length (WL), Prescaler Range (PSR), Prescaler Modulus (PM) and Frame rate (DC) selects the ratio of ccm_ssi_clk to sampling clock STFS. In case of I2S mode, the oversampling clock ccm_ssi_clk can be made available on this port if the SYS_CLK_EN bit is set. The relationship between the clocks and the dividers is shown in Figure 56-32 (“SSI Clocking”). The bit clock can be received from an SSI clock port or can be generated from the ccm_ssi_clk through a divider, as shown in Figure 56-33 (“SSI Transmit Clock Generator Block Diagram”).

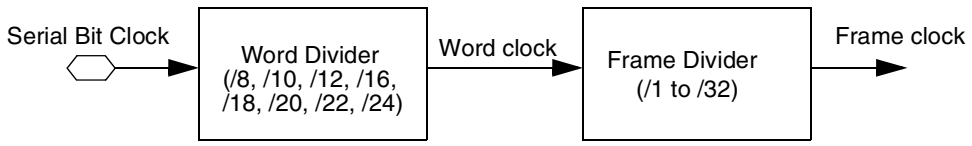


Figure 56-32. SSI Clocking

56.4.2.1 SSI Clock and Frame Sync Generation

Data clock and frame sync signals can be generated internally, or can be obtained from external sources. If internally generated, the SSI clock generator is used to derive bit clock and frame sync signals from the ccm_ssi_clk clock. The SSI clock generator consists of a selectable, fixed prescaler and a programmable prescaler for bit rate clock generation. In Gated Clock mode, the data clock is valid only when data is being transmitted. Otherwise the clock port is pulled to the inactive state. A programmable frame rate divider and a word length divider are used for frame rate sync signal generation.

Figure 56-33 shows a block diagram of the clock generator for the transmit section. The serial bit clock can be internal or external, depending on the Transmit Direction (TXDIR) bit in the SSI Transmit Configuration Register (STCR). The receive section contains an equivalent clock generator circuit.

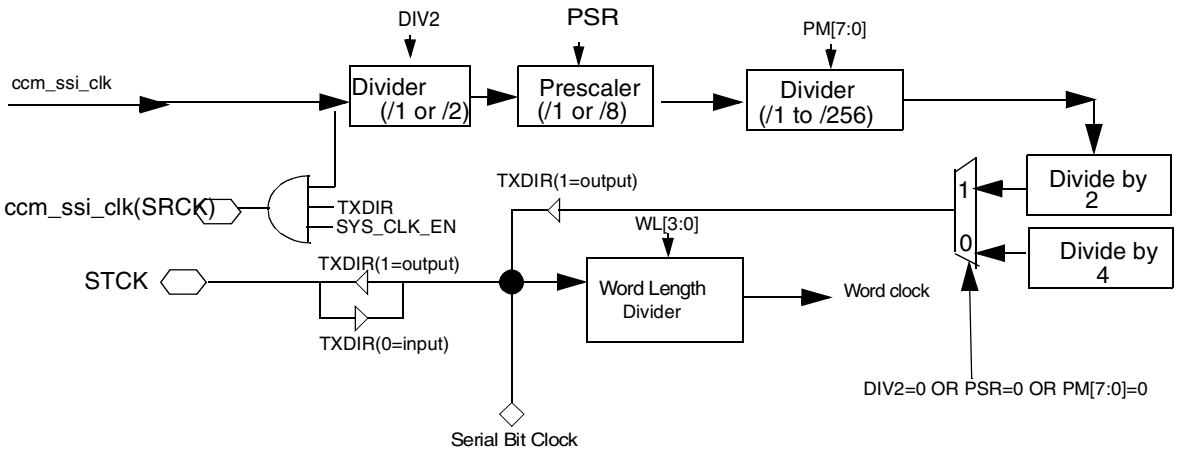


Figure 56-33. SSI Transmit Clock Generator Block Diagram

See Figure 56-34 shows the Frame Sync Generator block for the transmit section. When internally generated, both receive and transmit frame sync are generated from the word clock and are defined by the Frame Rate Divider (DC[4:0]) bits and the Word Length (WL[3:0]) bits of the SSI Transmit Clock Control Register (STCCR). The receive section contains an equivalent circuit for the Frame Sync Generator.

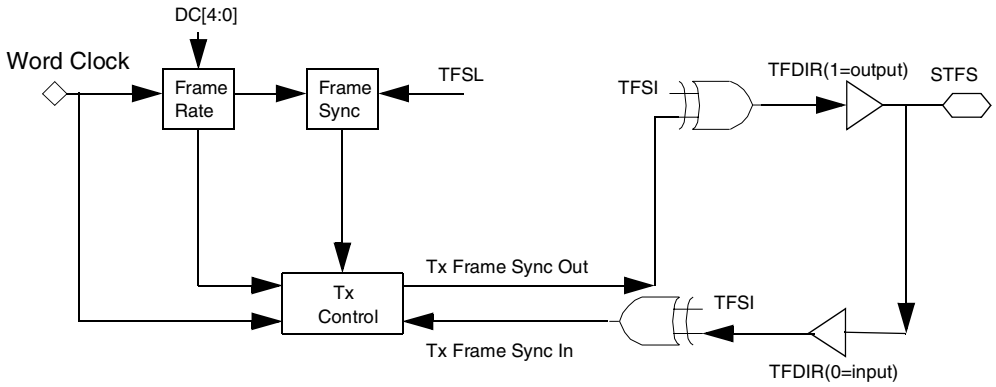


Figure 56-34. SSI Transmit Frame Sync Generator Block Diagram

56.4.2.2 DIV2, PSR and PM Bit Description

The bit clock frequency can be calculated from the SSI Serial System Clock (ccm_ssi_clk) using the equation in Figure 56-35.

NOTE

Ensure that the bit-clock frequency must be 5 times the ipg_clk frequency. The oversampling clock frequency can go up to ipg_clk frequency. Bits DIV2, PSR, and PM should not be all set to zero at the same time.

$$f_{INT\_BIT\_CLK} = f_{ccm\_ssi\_clk} / [(DIV2 + 1) \times (7 \times PSR + 1) \times (PM + 1) \times 2]$$

where PM=PM[7–0]

$$f_{FRAME\_SYN\_CLK} = (f_{INT\_BIT\_CLK}) / [(DC + 1) \times WL]$$

where DC = DC[4:0] and WL = 8, 10, 12, 16, 18, 20, 22, 24

Note: When DIV2 =0 and PSR = 0 and PM= 0, Frequency of bit_clk will be calculated as :-

$$f_{INT\_BIT\_CLK} = f_{ccm\_ssi\_clk} / 4$$

Figure 56-35. SSI Bit Clock Equation

For example, if the SSI oversampling clock (ccm_ssi_clk) is 12.288, in 8-bit word Normal mode with DC[4:0] set to 1 (00001), PM[7:0] set to 47 (0010 1111), the PSR bit cleared, DIV2 bit set to 1, a bit clock rate of 12.288 Mhz ÷ [1 × 4 × 48] = 64 kHz is generated. Since the 8-bit word rate is equal to one (i.e. normal mode), the sampling rate (FS rate) would then be 64 kHz ÷ [1 × 8] = 8 kHz.

In next example, the oversampling clock (ccm_ssi_clk) clock is 11.2896 Mhz. A 16-bit word Network mode with DC[4:0] set to 1 (00001), PM[7:0] set to 3 (0000 0011), the PSR bit is set to 0, DIV2 bit set to 0, and a 11.2896 MHz oversampling clock, a bit clock rate of 11.2896 Mhz ÷ [1 × 2 × 4] = 1.4112 MHz is generated. Since the 16-bit word rate is equal to two, the sampling rate (FS rate) would be 1.4112 MHz ÷ [2 × 16] = 44.1 kHz.

Table 56-31 shows programming examples for the clock dividers in the CRM and the SSI to support various bit clock (STCK) frequencies.

Table 56-31. SSI Bit Clock and Frame Rate as a Function of PSR, PM, and DIV2

Bits/ Word	Words/ Frame	Ideal Frame Rate (kHz)	PLL Freq (Mhz)	SSIDIV (in CRM)	MCLK/ ccm_ssi_clk Freq (Mhz)	DIV 2	PS R	PM	WL	DC	Actual Bit_Clk Freq (kHz) STCK	Target Bit_Clk Freq (kHz) STCK	Error (Hz)
16	1	8	294.912	48	12.288	0	0	47	7	0	128	128	0
16	2	8	294.912	48	12.288	0	0	23	7	1	256	256	0
16	4	8	294.912	48	12.288	0	0	11	7	3	512	512	0
16	1	12	294.912	48	12.288	0	0	31	7	0	192	192	0
16	2	12	294.912	48	12.288	0	0	15	7	1	384	384	0

Table 56-31. SSI Bit Clock and Frame Rate as a Function of PSR, PM, and DIV2

Bits/ Word	Words/ Frame	Ideal Frame Rate (kHz)	PLL Freq (Mhz)	SSIDIV (in CRM)	MCLK/ ccm_ssi_clk Freq (Mhz)	DIV 2	PS R	PM	WL	DC	Actual Bit_Clk Freq (kHz) STCK	Target Bit_Clk Freq (kHz) STCK	Error (Hz)
16	4	12	294.912	48	12.288	0	0	7	7	3	768	768	0
16	1	16	294.912	48	12.288	0	0	23	7	0	256	256	0
16	2	16	294.912	48	12.288	0	0	11	7	1	512	512	0
16	4	16	294.912	48	12.288	0	0	5	7	3	1024	1024	0
16	1	24	294.912	48	12.288	0	0	15	7	0	384	384	0
16	2	24	294.912	48	12.288	0	0	7	7	1	768	768	0
16	4	24	294.912	48	12.288	0	0	3	7	3	1536	1536	0
16	1	32	294.912	48	12.288	0	0	11	7	0	512	512	0
16	2	32	294.912	48	12.288	0	0	5	7	1	1024	1024	0
16	4	32	294.912	48	12.288	0	0	2	7	3	2048	2048	0
16	1	48	294.912	48	12.288	0	0	15	7	0	768	768	0
16	2	48	294.912	48	12.288	0	0	3	7	1	1536	1536	0
16	4	48	294.912	48	12.288	0	0	1	7	3	3072	3072	0
16	1	11.025	270.9504	48	11.2896	0	0	31	7	0	176.4	176.4	0
16	2	11.025	270.9504	48	11.2896	0	0	15	7	1	352.8	352.8	0
16	4	11.025	270.9504	48	11.2896	0	0	7	7	3	705.6	705.6	0
16	1	22.05	270.9504	48	11.2896	0	0	15	7	0	352.8	352.8	0
16	2	22.05	270.9504	48	11.2896	0	0	7	7	1	705.6	705.6	0
16	4	22.05	270.9504	48	11.2896	0	0	3	7	3	1411.2	1411.2	0
16	1	44.1	270.9504	48	11.2896	0	0	7	7	0	705.6	705.6	0
16	2	44.1	270.9504	48	11.2896	0	0	3	7	1	1411.2	1411.2	0
16	4	44.1	270.9504	48	11.2896	0	0	1	7	3	2822.4	2822.4	0

NOTE

Table 56-31 describes how various frame rates can be achieved with the PLL0/1 supplying a frequency of 294.912 MHz and 270.9504 MHz (with WL and DC settings as shown). Using PLL2 requires that these input frequencies be lowered by a factor of 2 (and the dividers be changed accordingly).

Note that using the MRCG allows the input frequency to be lowered by a factor of 4 (provided the dividers are changed accordingly). These clocks are recommended as convenient starting points, but the system allows other input clock frequencies as well.

Table 56-31 below shows programming of the CRM and SSI dividers in order to generate the appropriate oversampling clock and BIT_CLK frequencies for various sampling rates. In these examples, the master mode is selected either by setting I2S master bit (SCR[6:5]=01) or individually programming the SSI in network, synchronous, transmit internal mode (the table specifically illustrates the I2S mode frequencies/sample rates). The oversampling clock is ccm_ssi_clk.

Note that the I2S master mode requires that a word length of 32 bits be used (regardless of the actual data type). Consequently, the fixed I2S frame rate of 64 bits per frame (word length (WL) can be any value) and DC of 1 are assumed.

56.4.3 Receive Interrupt Enable Bit Description

When the RIE and RE bit are set, the processor is interrupted when either of the SSI Receive FIFO Full (RFF0/1) bits in SISR is set (if the corresponding Receive FIFO is enabled). If the Receive FIFO is not enabled, the interrupt is generated when the corresponding SSI Receive Data Ready (RDR0/1) bit in the SISR is set. When the receive FIFO is enabled, a maximum of 15 values are available to be read (15 values per channel in Two-Channel mode). If not enabled, then one value can be read from the SRX register (one each in case of Two-Channel mode). If the RIE bit is cleared, these interrupts are disabled. However, the RFF0/1 and RDR0/1 bits still indicate the receive data register full condition. Reading the SRX registers clears the RDR bits, thus clearing the pending interrupt. Two receive data interrupts (two per channel in case of Two-Channel mode) are available: receive data with exception status and receive data without exception. Table 56-32 and Table 56-33 show the conditions under which these interrupts are generated.

Table 56-32. SSI Receive Data 1 Interrupts

Interrupt	RIE	ROE0	RFF0/RDR0
Receive Data 1(with Exception Status)	1	1	1
Receive Data 1(without exception)	1	0	1

Table 56-33. SSI Receive Data 0 Interrupts

Interrupt	RIE	ROE1	RFF1/RDR1
Receive Data 0 (with Exception Status)	1	1	1
Receive Data 0 (without exception)	1	0	1

56.4.4 Transmit Interrupt Enable Bit Description

The SSI Transmit Interrupt Enable (TIE) control bit determines whether the processor is interrupted when the SSI transmitter needs to be serviced. When the TIE and TE bits are set, the program controller is interrupted when either of the SSI Transmit FIFO Empty (TFE0/1) flags in SISR are set (if corresponding Transmit FIFO is enabled). If the corresponding Transmit FIFO is not enabled, an interrupt is generated when the corresponding SSI Transmit Data Register Empty (TDE0/1) flag in the SISR is set and Transmit Enable (TE) bit is set.

When Transmit FIFO 0 is enabled, a maximum of 15 values can be written to the SSI (15 per channel in case of Two-Channel mode, using Tx FIFO 1). If not enabled, then one value can be written to the STX0 register (one per channel in case of Two-Channel mode using STX1). When the TIE bit is cleared, all transmit interrupts are disabled. However, the TDE0/1 bits always indicate the corresponding STX register empty condition, even when the transmitter is disabled by the Transmit Enable (TE) bit (in the SCR). Writing data to the STX clears the corresponding TDE bit, thus clearing the interrupt. Two transmit data interrupts are available (four in case of Two-Channel mode, two per channel): transmit data with exception

status and transmit data without exceptions. Table 56-34 and Table 56-35 show the conditions under which these interrupts are generated.

Table 56-34. SSI Transmit Data 1 Interrupts

Interrupt	TIE	TUE1	TFE1/TDE1
Transmit Data 1 (with Exception Status)	1	1	1
Transmit Data 1 (without exception)	1	0	1

Table 56-35. SSI Transmit Data 0 Interrupts

Interrupt	TIE	TUE0	TFE0/TDE0
Transmit Data 0 (with Exception Status)	1	1	1
Transmit Data 0 (without exception)	1	0	1

56.4.5 Internal Frame and Clock Shutdown

During transmit/receive operation, disabling TE/RE will ensure that data transmission/reception stops after current frame ends following which TFRC/RFRC Status bits will get set to indicate the Frame Completion State. If TE is disabled 4 clock cycles before the next frame, extra frame generated are invalid frames. TFR_CLK_DIS/RFR_CLK_DIS bit is set in the current or any of the previous frames, SSI will stop driving the STFS/SRFS and STCK/SRCK signals after the current frame ends.

If TFR_CLK_DIS/RFR_CLK_DIS bit is not set, SSI will continue generating STFS/SRFS and STCK/SRCK signals (in case direction is from SSI), which then can be disabled by writing ‘1’ to TFS_CLK_DIS/RFR_CLK_DIS bit. SSI will then stop driving these signals after end of frame is reached following which TFRC/RFRC status bits will get set to indicate the Frame Completion State.

Figure 56-36 is an illustration of transmission case where TXDIR and TFDIR are both set to ‘1’. In this case TE is disabled with TFS_CLK_DIS bit set in current or any of the previous frames.

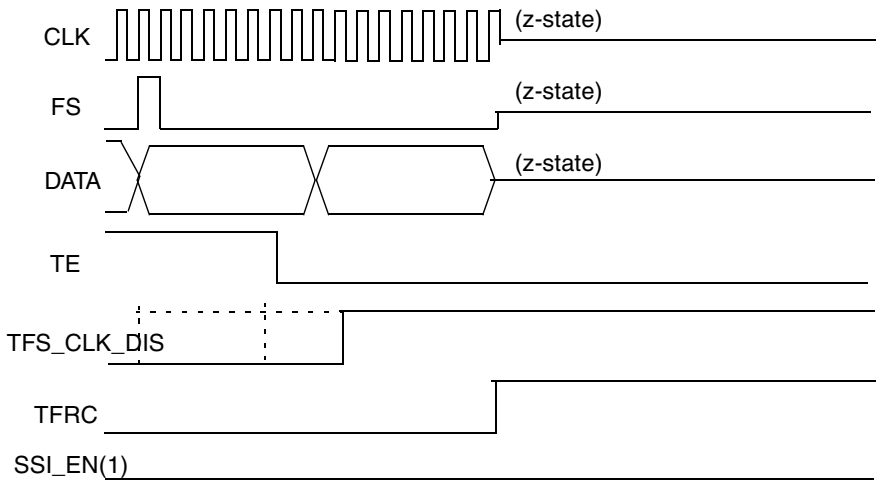


Figure 56-36. TFS_CLK_DIS assertion in current or previous frame as TE disable

Figure 56-37 is an illustration of transmission case where TXDIR and TFDIR are both set to ‘1’. In this case TFS_CLK_DIS bit is set after few frames of disabling TE. TFRC (Transmit Frame Complete) is set at frame boundary after TE is cleared. Once software services this interrupt and sets TFR_CLK_DIS bit later, TFRC bit is again set at next frame boundary.

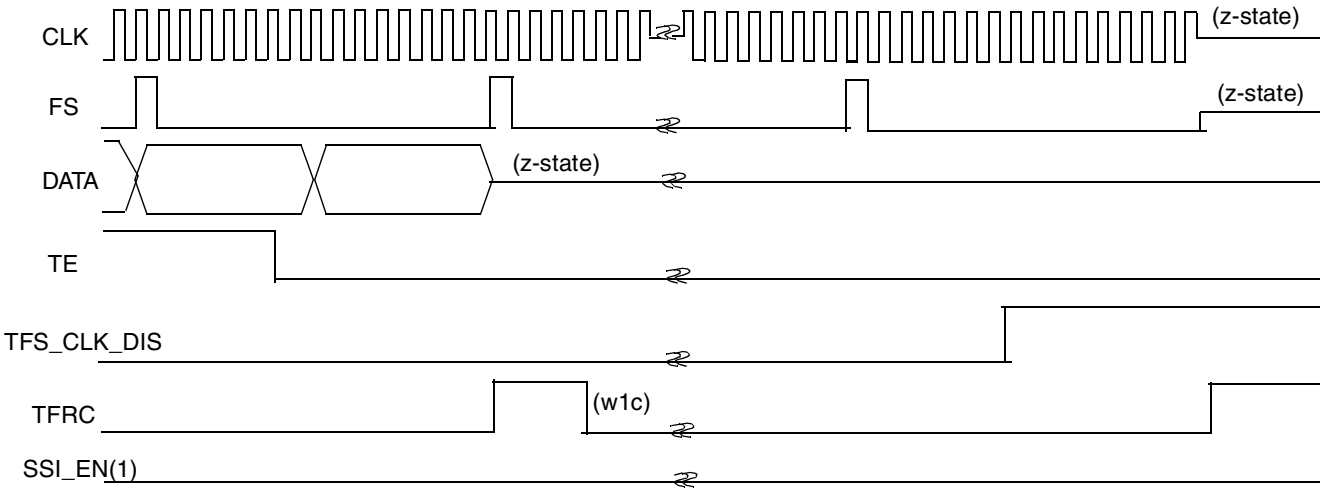


Figure 56-37. TFS_CLK_DIS assertion in subsequent frame after disabling TE

56.4.6 IP Bus Interface

The SSI has an IP Bus interface compliant with SRS 3.0.2 in order to provide a control and data interface. This interface is used by both the processor and DMA controller.

56.4.6.1 Transfer Lengths Supported

The IP Bus interface of the SSI only supports 32-bit transfers with all SSI registers other than STX0, STX1, SRX0, and SRX1 (that is, the data registers). With the exception of the data registers, using 8-bit and 16-bit transactions could result in undesired behavior but will not result in a transfer bus error. The data registers (STX0, STX1, SRX0, and SRX1) support 8-bit, 16-bit, and 32-bit transfer lengths without restrictions.

56.4.6.2 Transfer Bus Errors

Transfer bus errors are generated upon response to the following:

- Write transfer to a read-only register.
- Read or write access to a register space beyond the last populated register of the SSI in its memory map (up until the end of the allocated memory address range of the SSI).

56.4.6.3 Clock Rate

The IP Bus clock frequency must be at least five times the serial bit clock frequency.

56.5 Initialization/Application Information

The SSI is affected by the following types of reset:

- Power-on Reset—The Power-on reset is generated by asserting the RESET port. The Power-on reset clears the SSIEN bit in SCR, which disables the SSI. All other status and control bits in the SSI are affected as described in SSI Programming Model in Section 56.3.
- SSI Reset—The SSI reset is generated when the SSIEN bit in the SCR is cleared. The SSI status bits are preset to the same state produced by the Power-on reset. The SSI control bits are unaffected. The control bits in the SCR are also unaffected. The SSI reset is useful for selective reset of the SSI without changing the present SSI control bits and without affecting the other peripherals.

The correct sequence to initialize the SSI is as follows:

1. Issue a Power-on or SSI reset (SCR[SSIEN]=0).
2. Disable SSI clocks (ipg_clk, ccm_ssi_clk).
3. Set all control bits for configuring the SSI (see Table 56-36 for the list of “SSI Control Bits Requiring SSI to be Disabled Before Change”).
4. Enable appropriate interrupts/DMA requests through SIER.
5. Set the SCR[SSIEN] bit (=1) to enable the SSI.
6. Enable SSI clocks (ipg_clk, ccm_ssi_clk), as required.
7. In case of AC97 mode, set the SACNT[AC97EN] bit after programming the SATAG register (if needed, for AC97 Fixed mode).
8. In case of gated mode of operation refer Table 56-5.
9. Set SCR[TE/RE] bits.
10. To ensure proper operation of the SSI, use the “Power-on or SSI reset before changing any of the SSI Control” bits listed in Table 56-36).

NOTE

These control bits should not be changed when SSI is enabled

Table 56-36. SSI Control Bits Requiring SSI to be Disabled Before Change

Control Register	Bit
SCR	[9]=CLK_IST [8]=TCH_EN [7]=SYS_CLK_EN [6:5]=I2S_MODE [4]=SYN [3]=NET
SIER	[22]=RDMAE [20]=TDMAE

Table 56-36. SSI Control Bits Requiring SSI to be Disabled Before Change (continued)

Control Register	Bit
SRCR STCR	[9]=RXBIT0 [9]=TXBIT0 [8]=RFEN1 [8]=TFEN1 [7]=RFEN0 [7]=TFEN0 [6]=RFDIR [6]=TFDIR [5]=RXDIR [5]=TXDIR [4]=RSHFD [4]=TSHFD [3]=RSCKP [3]=TSCKP [2]=RFSI [2]=TFSI [1]=RFSL [1]=TFSL [0]=REFS [0]=TEFS
SRCCR STCCR	[16]=WL3 [15]=WL2 [14]=WL1 [13]=WL0
SACNT	[1]=FV [10:5]=FRDIV
PHCONFIG	[10:7]=CLKSRCSEL [0:2]=GAINSEL

Chapter 57
TrustZone Interrupt Controller (TZIC)

57.1 Introduction

The TZIC is a TrustZone enabled interrupt controller with an AXI interface. It allows complete and independent control over every interrupt connected to the controller, including enable, security, and priority functionality. It provides secure and non-secure transaction access to those interrupts, restricting non-secure read and write transactions to only those interrupts configured as non-secure and allowing secure transactions read and write capability to all interrupts regardless of security configuration. It supports priority masking, overall enable/disable of secure or nonsecure interrupts, and access to raw interrupt state and pending transactions.

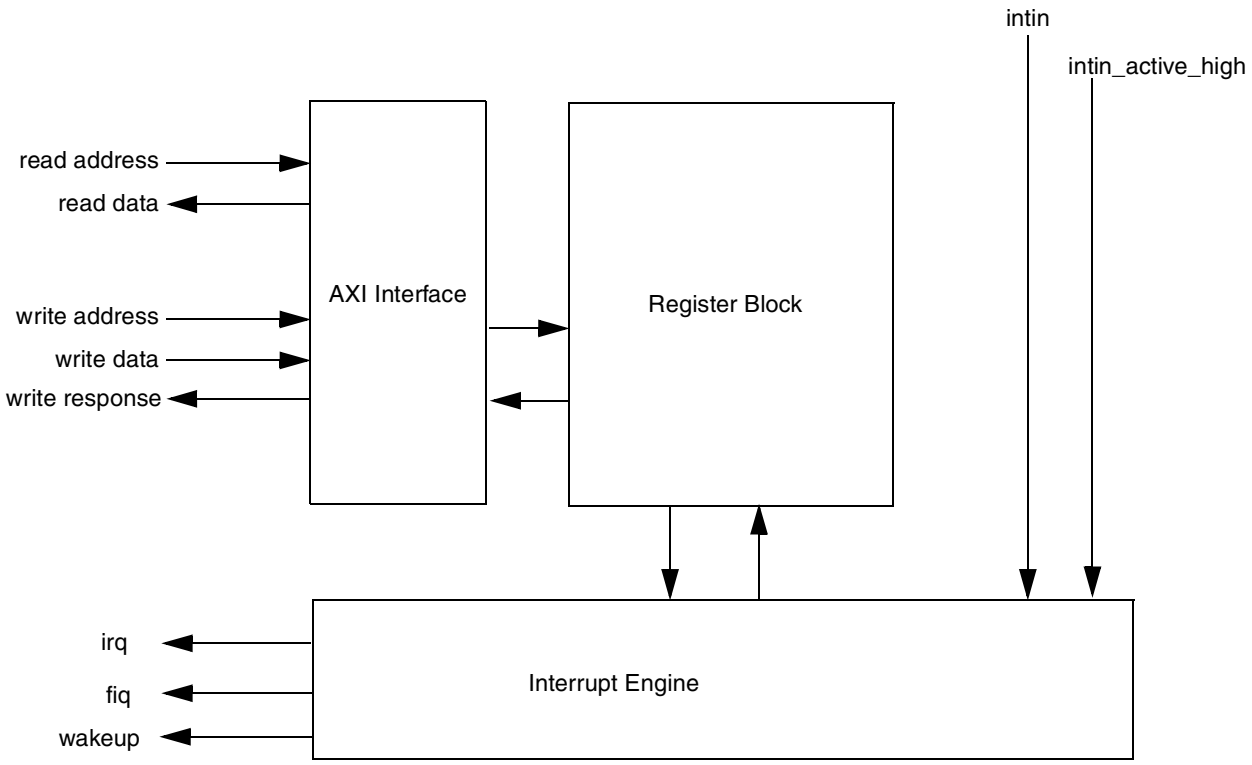


Figure 57-1. Block Diagram

57.1.1 Features

The following features are included in the TZIC:

- AXI interface
- Support for up to 128 interrupts
- Secure configuration available on all interrupts
- Software triggered interrupts supported
- High priority pending registers for autovectored interrupt service routines
- Set/Clear registers for interrupt enable, force, and secure registers
- 16/32 programmable priority levels (nonsecure/secure)
- Wakeup configuration available on all interrupts

57.1.2 Modes of Operation

The TZIC has one mode of operation. Out of reset all configurations are available.

57.2 External Signal Description

The TZIC has no external signals.

57.3 Memory Map and Register Definition

This section contains a memory map, register key, table of register conventions, and register summary.

57.3.1 TZIC Register Memory Map

Table 57-1 shows the memory map for all possible registers in the TZIC. The presence of some registers is dependent upon the implementation configuration. Unspecified address spaces are reserved.

Table 57-1. TZIC Memory Map

Address	Registers	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x0000 (INTCTRL)	Control Register	R/W	0x0000_0000	57.3.3.2/57-9
0xBASE+0x0004 (INTTYPE)	Interrupt Controller Type Register	R	0x0000_0403	57.3.3.3/57-11
0xBASE+0x000C (PRIOMASK)	Priority Mask Register	R/W	0x0000_0000	57.3.3.4/57-12
0xBASE+0x0010 (SYNCCTRL)	Synchronizer Control	R/W	0x0000_0000	57.3.3.5/57-14
0xBASE+0x0014 (DSMINT)	DSM Interrupt Holdoff	R/W	0x0000_0000	57.3.3.6/57-14

Table 57-1. TZIC Memory Map (continued)

Address	Registers	Access	Reset Value	Section/Page
0xBASE+0x0080 (INTSEC0)	Interrupt Security Register: Irq 0 to 31	R/W	0x0000_0000	57.3.3.7/57-15
...	...	...	...	
0xBASE+0x008C (INTSEC3)	Interrupt Security Register: Irq 96 to 127	R/W	0x0000_0000	57.3.3.7/57-15
0xBASE+0x0100 (ENSET0)	Enable Set Register: Irq 0 to 31	R/W	0x0000_0000	57.3.3.8/57-16
...	...	...	...	
0xBASE+0x010C (ENSET3)	Enable Set Register: Irq 96 to 127	R/W	0x0000_0000	57.3.3.8/57-16
0xBASE+0x0180 (ENCLEAR0)	Enable Clear Register: Irq 0 to 31	R/W	0x0000_0000	57.3.3.9/57-17
...	...	...	...	
0xBASE+0x010C (ENCLEAR3)	Enable Clear Register: Irq 96 to 127	R/W	0x0000_0000	57.3.3.9/57-17
0xBASE+0x0200 (SRCSET0)	Source Set Register: Irq 0 to 31	R/W	0x0000_0000	57.3.3.10/57-18
...	...	...	...	
0xBASE+0x020C (SRCSET3)	Source Set Register: Irq 96 to 127	R/W	0x0000_0000	57.3.3.10/57-18
0xBASE+0x0280 (SRCCLEAR0)	Source Clear Register: Irq 0 to 31	R/W	0x0000_0000	57.3.3.11/57-20
...	...	...	...	
0xBASE+0x028C (SRCCLEAR3)	Source Clear Register: Irq 96 to 127	R/W	0x0000_0000	57.3.3.11/57-20
0xBASE+0x0400 (PRIORITY0)	Priority Register: Irq 0 to 3	R/W	0x0000_0000	57.3.3.12/57-21
...	...	...	...	
0xBASE+0x047C (PRIORITY31)	Priority Register: Irq 124 to 127	R/W	0x0000_0000	57.3.3.12/57-21
0xBASE+0x0D00 (PND0)	Pending Register: Irq 0 to 31	R	0x0000_0000	57.3.3.13/57-23
...	...	...	...	
0xBASE+0x0D0C (PND3)	Pending Register: Irq 96 to 127	R	0x0000_0000	57.3.3.13/57-23
0xBASE+0x0D80 (HIPND0)	High Priority Pending Register: Irq 0 to 31	R	0x0000_0000	57.3.3.14/57-24
...	...	...	...	

Table 57-1. TZIC Memory Map (continued)

Address	Registers	Access	Reset Value	Section/Page
0xBASE+0x0D8C (HIPND3)	High Priority Pending Register: Irq 96 to 127	R	0x0000_0000	57.3.3.14/57-24
0xBASE+0x0E00 (WAKEUP0)	Wakeup Config Register: Irq 0 to 31	R	0x0000_0000	57.3.3.15/57-25
...	...	...	...	
0xBASE+0x0E0C (WAKEUP3)	Wakeup Config Register: Irq 96 to 127	R	0x0000_0000	57.3.3.15/57-25
0xBASE+0x0F00 (SWINT)	Software Interrupt Trigger Register	W	0x0000_0000	57.3.3.16/57-26

57.3.2 Register Summary

The conventions in Figure 57-2 and Table 57-2 serve as a key for the register summary and individual register diagrams.

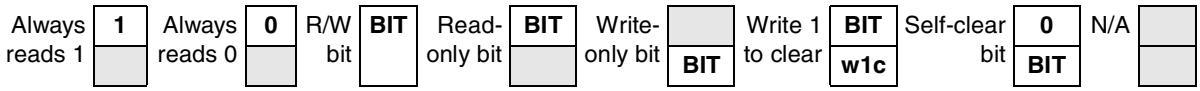


Figure 57-2. Key to Register Fields

Table 57-2 provides a key for register figures and tables and the register summary.

Table 57-2. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.

Table 57-2. Register Conventions (continued)

Convention	Description
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 57-3 shows the register summary for the TZIC.

Table 57-3. TZIC Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0000 (INTCTRL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NSEN
	W	NSEN MASK															
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EN
	W																
0xBASE+0x0004 (INTTYPE)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	DOM	0	0	CPUS			ITLINES				
	W																
0xBASE+0x000C (PRIOMASK)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	MASK							
	W																
0xBASE+0x0010 (SYNCCTRL)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SYNCMODE	
	W																
0xBASE+0x0014 (DSMINT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DSM
	W																
0xBASE+0x0080 (INTSEC0)	R	SECURE[31:16]															
	W																
	R	SECURE[15:0]															
	W																

Table 57-3. TZIC Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x008C (INTSEC3)	R	SECURE[31:16]															
	W																
	R	SECURE[15:0]															
	W																
0xBASE+0x0100 (ENSET0)	R	INTENSET[31:16]															
	W																
	R	INTENSET[15:0]															
	W																
0xBASE+0x010C (ENSET3)	R	INTENSET[31:16]															
	W																
	R	INTENSET[15:0]															
	W																
0xBASE+0x0180 (ENCLEAR0)	R	INTENCLEAR[31:16]															
	W																
	R	INTENCLEAR[15:0]															
	W																
0xBASE+0x010C (ENCLEAR3)	R	INTENCLEAR[31:16]															
	W																
	R	INTENCLEAR[15:0]															
	W																
0xBASE+0x0200 (SRCSET0)	R	SRCSET[31:16]															
	W																
	R	SRCSET[15:0]															
	W																
0xBASE+0x020C (SRCSET3)	R	SRCSET[31:16]															
	W																
	R	SRCSET[15:0]															
	W																
0xBASE+0x0280 (SRCCLEAR0)	R	SRCCLEAR[31:16]															
	W																
	R	SRCCLEAR[15:0]															
	W																

Table 57-3. TZIC Register Summary (continued)

Name		31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0xBASE+0x028C (SRCCLEAR3)	R	SRCCLEAR[31:16]
	W	
	R	SRCCLEAR[15:0]
	W	
0xBASE+0x0400 (PRIORITY0)	R	PRIO3 PRIO2
	W	
	R	PRIO1 PRIO0
	W	
0xBASE+0x047C (PRIORITY31)	R	PRIO3 PRIO2
	W	
	R	PRIO1 PRIO0
	W	
0xBASE+0x0D00 (PND0)	R	PND[31:16]
	W	
	R	PND[15:0]
	W	
0xBASE+0x0D0C (PND3)	R	PND[31:16]
	W	
	R	PND[15:0]
	W	
0xBASE+0x0D80 (HIPND0)	R	HIPND[31:16]
	W	
	R	HIPND[15:0]
	W	
0xBASE+0x0D8C (HIPND3)	R	HIPND[31:16]
	W	
	R	HIPND[15:0]
	W	
0xBASE+0x0E00 (WAKEUP0)	R	WAKEUP[31:16]
	W	
	R	WAKEUP[15:0]
	W	

Table 57-3. TZIC Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0E0C (WAKEUP3)	R	WAKEUP[31:16]															
	W																
	R	WAKEUP[15:0]															
	W																
0xBASE+0x0F00 (SWINT)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W	INT NEG															
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W							INTID									

57.3.3 Register Descriptions

This section contains the detailed register descriptions for the TZIC registers.

The read and write behavior of many of the TZIC registers is impacted by the security configuration of each individual interrupt, as determined by the INTSEC registers. In all cases, non-secure transactions will only read back information about interrupts configured as non-secure and writes will have no effect on secure interrupts. Secure transactions will have full read and write access to all interrupt bits.

Write bursts are taken with one clock per write data after an initial one clock delay. Read bursts are returned with two clocks per read data.

The specific behavior for each register set is described in the field descriptions. In some cases where the register described spans multiple 32-bit addressable locations, the variable X is used to denote which particular register corresponds to which interrupts. For instance, a register with 1 bit associated with each interrupt will have one 32-bit register for each set of 32 interrupts. The registers can be numbered such that REGISTER0 corresponds to interrupts 0–31, REGISTER1 corresponds to interrupts 32–63, and so on. This relationship can be expressed using the following equation:

REGISTER(X)[Y] corresponds to interrupt $(32 \times (X))+Y$

Such that if X is 2 and Y is 17:

REGISTER(2)[17] $\rightarrow (32 \times (2))+17 = 81$

The interrupt would be number 81.

This terminology is used to express the location and relationship for several registers in this section.

57.3.3.1 Supported Register Transaction Types

All registers are accessible with 32-bit accesses. The PRIOMASK register and PRIORITY registers support 8-bit transactions; all others are not supported and yield unpredictable results. 16-bit transactions

are not supported to any registers and result in a slave error. Any access larger than 32-bit also results in an error.

All registers are bufferable, non-cacheable, data, supervisor only. Transactions marked as cacheable, allocate, user, or instruction result in error responses.

57.3.3.2 Control Register

The interrupt control register (INTCTRL), shown in Figure 57-3, can enable or disable all normal or secure interrupts to the CPU, depending on the state of the write transaction used. A secure read or write accesses the enable bit for secure interrupts, and a non-secure read or write accesses the enable bit for non-secure interrupts. This only affects the outputs to the core; it has no impact on which interrupts are pending in the PND and HIPND registers.

A secure read can access the non-secure enable through bit 16, which is only written if bit 31 is written at the same time. This allows secure software to ignore the non-secure enable bit and update the secure enable bit without a read-modify-write sequence.

This register can only be accessed by a 32-bit supervisor transaction.

0xBASE+0x0000 (INTCTRL)

Access: Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	NSEN
W	NSENMASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-3. Interrupt Control Register

Table 57-4 shows the INTCTRL field descriptions.

Table 57-4. INTCTRL Field Descriptions

Field	Description
31 NSEN MASK	Non-Secure Enable Mask This bit indicates the secure write transaction is updating the NSEN bit. When written as 1'b0 the NSEN bit will not be updated. When written as 1'b1 the NSEN bit will be updated. This allows the secure software to ignore the NSEN bit and update only the Secure EN bit without requiring a read/modify/write to determine the current state of the NSEN. If written with AWPROT[1] == 1'b0: 0 NSEN bit unaffected 1 NSEN bit updated If written with AWPROT[0[== 1'b1: no effect **Note:** Available for secure transactions only.
30–17	Reserved
16 NSEN	Non-Secure Enable This bit is available in secure mode only to enable/disable nonsecure interrupts. This allows for using both security levels in a single security level system where only secure transactions are available. A secure write to this bit will only take effect if the NSEN MASK bit is also set. If written or read with AxPROT[1] == 1'b0: 0 Non-Secure interrupts disabled 1 Non-Secure interrupts enabled If written or read with AxPROT[1] == 1'b1: On reads: 1'b0 On writes: no effect **Note:** Available for secure transactions only.
15–1	Reserved
0 EN	Interrupt Enable. This bit, when set, enables normal or secure interrupts, depending on the secure state of the transaction. If written or read with AxPROT[1] == 1'b0: 0 Secure interrupts disabled 1 Secure interrupts enabled If written or read with AxPROT[1] == 1'b1: 0 Non-secure interrupts disabled 1 Non-secure interrupts enabled

57.3.3.3 Interrupt Controller Type Register

The interrupt controller type register (INTTYPE), shown in Figure 57-4, contains information about the type of interrupt controller that has been implemented. It can be used to determine the number of interrupts, CPUs, and security domains enabled for a given instantiation.

The current version only supports configurations with one CPU and two security domains.

This register can only be accessed by a 32-bit supervisor read transaction.

0xBASE+0x0004 (INTTYPE) Access: Privileged Read Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	DOM	0	0	CPUS				ITLINES			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-4. Interrupt Controller Type Register

Table 57-5 shows the INTTYPE field descriptions.

Table 57-5. INTTYPE Field Descriptions

Field	Description
31–11	Reserved
10 DOM	Domains. This bit indicates the number of domains supported by the current configuration. 0 Supports 1 Security Domain 1 Supports 2 Security Domains
9–8	Reserved

Table 57-5. INTTYPE Field Descriptions

Field	Description
7–5 CPUS	CPU Count. This field indicates the number of CPUs supported by the current configuration. Currently only 1 CPU is supported, this field is specified for future compatability. 000 Supports 1 CPU 001 Supports 2 CPUs 010 Supports 3 CPUs ... 111 Supports 8 CPUs
4–0 ITLINES	Interrupt Lines. This field indicates the total number of interrupt lines supported by the current configuration. 00000 up to 32 interrupt lines supported 00001 up to 64 interrupt lines supported 00010 up to 96 interrupt lines supported ... 11110 up to 992 interrupt lines supported 11111 up to 1020 interrupt lines supported

57.3.3.4 Priority Mask Register

The Priority Mask Register (PRIOMASK), shown in Figure 57-5, is a single register used to mask both secure and non-secure interrupts based on their priority configuration. The CPU will only receive an interrupt if there is a pending interrupt with a priority higher than the value in this register. If an interrupt source asserts and its priority is equal to or lower than the priority mask value, it will not cause an interrupt to the CPU.

The lowest priority the priority mask value can be set to is the lowest priority level in the controller. Therefore any interrupt set to the lowest allowed priority will never cause an interrupt since it is not possible for its priority to be greater than the mask value.

This register is affected by an integration parameter which varies the available resolution of priority values. The number of bits available can vary from four to eight, with the lowest resolution only implementing bits [7:4], and the highest resolution implementing bits [7:0]. Which bits are implemented can be determined by writing 8’hFF to bits [7:0] and reading back the result. The bits set high will indicate which register bits are implemented in this register as well as the PRIORITY registers.

Secure and non-secure transactions to this register will have different behaviors. A secure write will result in the register receiving the write data bits as written. A non-secure write will results in wdata bits [7:1] being shifted to the right one bit and bit [7] being set to 1’b1. A secure read will return the actual value stored in the register while a non-secure read will return the register value shifted to the left one bit and bit [0] being filled with a 1’b0. This is subject to the actual register bits implemented according to the integration parameter.

For more information regarding priorities, see Section 57.4.1.1, TrustZone and Interrupt Priority”.

This register can only be accessed by 8-bit or 32-bit supervisor transactions.

0xBASE+0x000C (PRIOMASK)

Access: Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	MASK							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-5. Priority Mask Register

Table 57-6 shows the PRIOMASK field descriptions.

Table 57-6. PRIOMASK Field Descriptions

Field	Description
31–8	Reserved
7–0 MASK	Priority Mask - This field stores the priority mask value. Interrupts with a priority level set equal to or below this value are masked. For READS: If ARPROT[1] == 1'b0 (secure) 00000000 Priority mask is set to 0 00000001 Priority mask is set to 1 ... 11111111 Priority mask is set to 255 If ARPROT[1] == 1'b1 (nonsecure) 00000000 Priority mask is set to 128 00000010 Priority mask is set to 129 ... 11111110 Priority mask is set to 255 For WRITES: If AWPROT[1] == 1'b0 (secure) 00000000 Priority mask is set to 0 00000001 Priority mask is set to 1 ... 11111111 Priority mask is set to 255 If AWPROT[1] == 1'b1 (nonsecure) 00000000 Priority mask is set to 128 00000001 Priority mask is set to 128 00000010 Priority mask is set to 129 ... 11111110 Priority mask is set to 255 11111111 Priority mask is set to 255

57.3.3.5 Synchronizer Control

The Synchronizer Control Register (SYNCCTRL), shown in Figure 57-6, is used to control latency and power usage in the interrupt synchronizers. There are three configurations for synchronization:

- Low Latency—The synchronizer is clocked continuously and interrupts are synchronized as they arrive.
- Low Power—The synchronizer is not clocked until a difference is detected between the inputs and outputs of the synchronizer. When a difference is detected, the detection result is synchronized and used as a clock enable for the synchronizer, which introduces additional latency. Total latency is four clocks—two for the enable and two for the interrupt.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE+0x0010 (SYNCCTRL)

Access: Secure Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SYNCMODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-6. Synchronizer Control Register

Table 57-7 shows the SYNCCTRL field descriptions.

Table 57-7. SYNCCTRL Field Descriptions

Field	Description
31–2	Reserved
1–0 SYNCMODE	Synchronizer Mode. 00 Low Latency 01 Low Power 10 Reserved 11 Reserved

57.3.3.6 DSM Interrupt Holdoff Register

The DSM Interrupt Holdoff Register (DSMINT), shown in Figure 57-7, is used to stop interrupts to the CPU from occurring until a wakeup signal occurs. This is used for entering Deep Sleep Mode. When written, the DSM bit will cause the interrupt synchronizer to stop updating. If an interrupt occurs while this register is being written the bit will not be set and the CPU will have to try to write again. If no interrupt to the CPU occurs while the register is being written the write will succeed and the interrupt synchronizers

will not update until the DSM bit clears. The DSM bit is cleared by either a write to the DSMINT register or assertion of the dsm_wakeup_request output of the TZIC. When a wakeup request is asserted, the TZIC will remove the DSM interrupt holdoff bit and interrupts will resume.

This register can only be accessed by 32-bit secure supervisor transactions.

0xBASE+0x0014 (DSMINT)

Access: Secure Privileged Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DSM
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-7. DSM Interrupt Holdoff Register

Table 57-8 shows the DSMINT field descriptions.

Table 57-8. DSMINT Field Descriptions

Field	Description
31–1	Reserved
0 DSM	DSM Interrupt Holdoff 0 Interrupt synchronizer updates 1 Interrupt synchronizer does not update

57.3.3.7 Interrupt Security Registers

The Interrupt Security Registers (INTSEC), shown in Figure 57-8, are used to set interrupts as secure or non-secure. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, if its secure bit is set to 1’b0, the interrupt is secure and will cause a secure interrupt to the CPU. When set as secure, a particular interrupt will also be invisible to non-secure accesses, its configuration hidden during non-secure reads. If its bit is set to 1’b1, the interrupt is non-secure and will cause a non-secure interrupt to the CPU.

The values read in the INTSEC register represent the interrupts currently set as non-secure. Out of reset, all interrupts are configured as secure, the intention being that when the TZIC is used in a system that does

not use TrustZone, the secure bit of ARPROT/AWPROT will be tied to secure and the TZIC will act as always in secure mode.

These registers can only be accessed by 32-bit secure supervisor transactions.

0xBASE+0x0080 (INTSEC0)
0xBASE+0x0084 (INTSEC1)
0xBASE+0x0088 (INTSEC2)
0xBASE+0x008C (INTSEC3)

Access: Privileged
Secure
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SECURE[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SECURE[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-8. Interrupt Security Registers

Table 57-9 shows the INTSEC field descriptions.

Table 57-9. INTSEC Field Descriptions

Field	Description
31–0 SECURE	Interrupt Secure Status. When read, this register returns the secure bit values of the associated interrupts. When written, this register sets the configuration for the associated interrupts. The corresponding interrupts for register INTSECX are determined through the following formula: SECUREX[31] → (32 × X) + 31 SECUREX[0] → (32 × X) + 0 0 Interrupt is secure 1 Interrupt is non-secure

57.3.3.8 Enable Set Registers

The Enable Set Registers (ENSET), shown in Figure 57-9, are used to enable interrupt requests to the core. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, when read, if its enable bit is set to 1’b1, the interrupt is enabled and will be transmitted to the CPU. If its bit is set to 1’b0, any activity on that interrupt will have no effect.

Writing a bit to 1’b1 in this register will enable the interrupt associated with that bit. Writing a bit to 1’b0 will have no effect. To clear an enable bit, the Enable Clear Registers must be used.

The values read in the ENSET register represent the currently enabled interrupts.

These registers can only be accessed by 32-bit supervisor transactions.

0xBASE+0x0100 (ENSET0)
0xBASE+0x0104 (ENSET1)
0xBASE+0x0108 (ENSET2)
0xBASE+0x010C (ENSET3)

Access: Privileged Read-Write

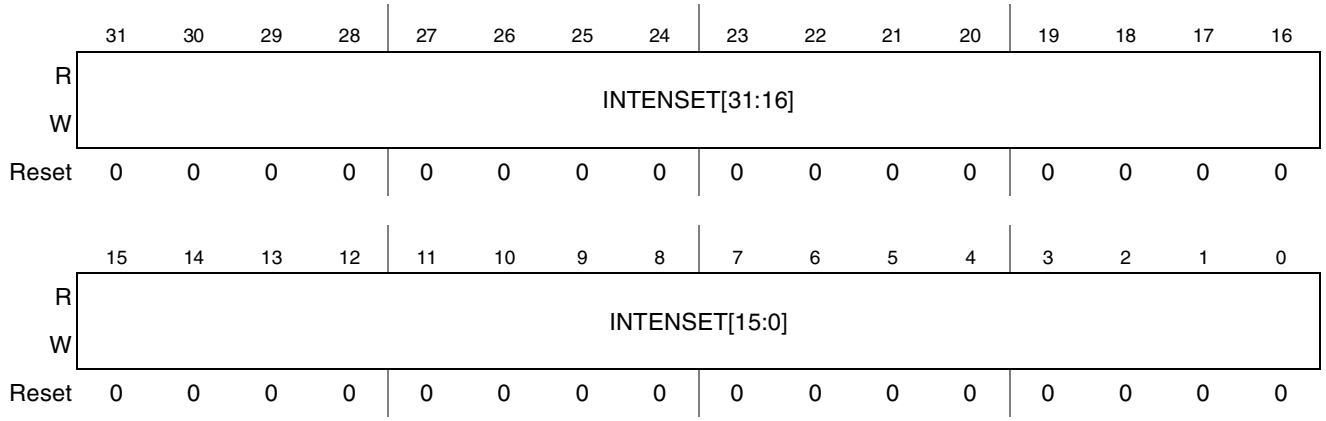


Figure 57-9. Interrupt Enable Set Registers

Table 57-10 shows the ENSET field descriptions.

Table 57-10. ENSET Field Descriptions

Field	Description
31–0 INTENSET	Interrupt Enable Set. When written, this register sets the enable bits of the associated interrupts. When read, this register returns the enable bit values of the associated interrupts. The corresponding interrupts for register ENSETX are determined through the following formula: INTENSET[31] → (32 × X) + 31 INTENSET[0] → (32 × X) + 0 If ARPROT[1] == 1'b0 (secure) 0 Interrupt is disabled 1 Interrupt is enabled If ARPROT[1] == 1'b1 (nonsecure) 0 Interrupt is disabled or configured as secure 1 Interrupt is enabled If AWPROT[1] == 1'b0 (secure) 0 no effect 1 Interrupt enable bit set If AWPROT[1] == 1'b1 (nonsecure) 0 no effect 1 Interrupt enable bit set if bit is configured as non-secure

57.3.3.9 Enable Clear Registers

The Enable Clear Registers (ENCLEAR), shown in Figure 57-10, are used to disable interrupt requests to the core. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, when read, if its enable bit is set to 1'b1, the interrupt is enabled and will be transmitted to the CPU. If its bit is set to 1'b0, any activity on that interrupt will have no effect.

Writing a bit to 1’b1 in this register will disable the interrupt associated with that bit. Writing a bit to 1’b0 will have no effect. To set an enable bit, the Enable Set Registers must be used.

The values read in the ENCLEAR register represent the currently enabled interrupts.

These registers can only be accessed by 32-bit supervisor accesses.

0xBASE+0x0180 (ENCLEAR0)
0xBASE+0x0184 (ENCLEAR1)
0xBASE+0x0188 (ENCLEAR2)
0xBASE+0x010C (ENCLEAR3)

Access: Privileged
Read-Write

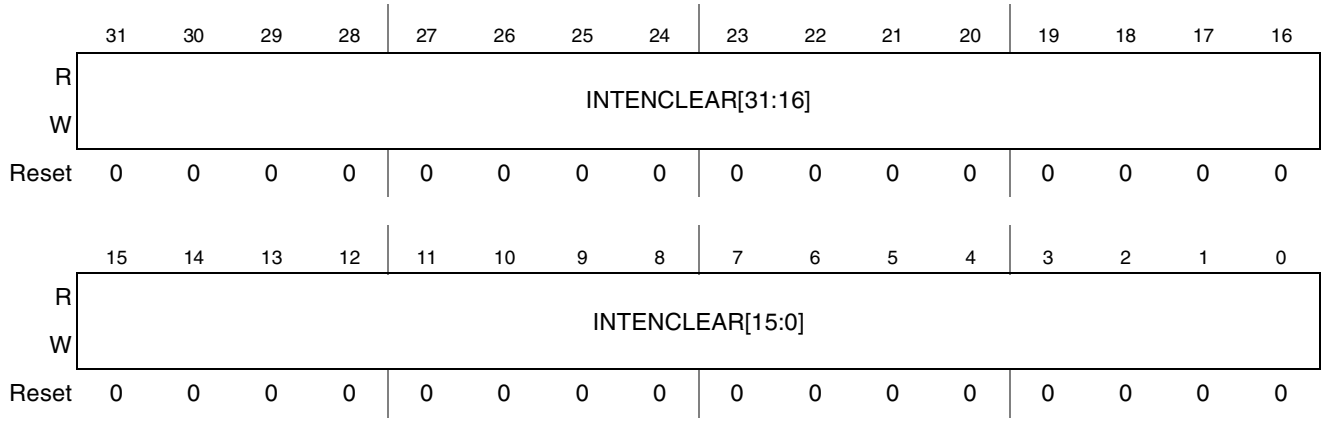


Figure 57-10. Interrupt Enable Clear Register

Table 57-11 shows the ENCLEAR field descriptions.

Table 57-11. ENCLEAR Field Descriptions

Field	Description
31-0 INTENCLEAR	Interrupt Enable Cleared. When written, this register clears the enable bits of the associated interrupts. When read, this register returns the enable bit values of the associated interrupts. The corresponding interrupts for register ENCLEARX are determined through the following formula: INTENCLEAR[31] → (32 × X) + 31 INTENCLEAR[0] → (32 × X) + 0 If ARPROT[1] == 1’b0 (secure) 0 Interrupt is disabled 1 Interrupt is enabled If ARPROT[1] == 1’b1 (nonsecure) 0 Interrupt is disabled or configured as secure 1 Interrupt is enabled If AWPROT[1] == 1’b0 (secure) 0 no effect 1 Interrupt enable bit cleared If AWPROT[1] == 1’b1 (nonsecure) 0 no effect 1 Interrupt enable bit cleared if bit is configured as non-secure

57.3.3.10 Source Set Registers

The Source Set Registers (SRCSET), shown in Figure 57-11, are used to determine which interrupts are currently asserted and to force interrupts to an asserted state regardless of activity on the interrupt lines.

Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, when read, if its source bit is set to 1’b1, the interrupt is asserted and pending action from the CPU. If its bit is set to 1’b0, the interrupt is not asserted.

Writing a bit to 1’b1 in this register will override the interrupt line and force the corresponding interrupt to assert. Writing a bit to 1’b0 will have no effect. To clear a source bit, the Source Clear Registers or SWINT must be used.

If a source bit is set which corresponds to an interrupt not enabled in the ENSET/ENCLEAR registers, the source bit will be 1’b1 but will not be considered pending and will not cause an interrupt to the core.

The values read in the SRCSET register represent the currently asserted interrupts.

These registers can only be accessed by 32-bit supervisor accesses.

0xBASE+0x0200 (SRCSET0)
0xBASE+0x0204 (SRCSET1)
0xBASE+0x0208 (SRCSET2)
0xBASE+0x020C (SRCSET3)

Access: Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRCSET[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRCSET[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-11. Interrupt Pending Set Registers

Table 57-12 shows the SRCSET field descriptions.

Table 57-12. SRCSET Field Descriptions

Field	Description
31–0 SRCSET	Interrupt Source Set. When written, this register forces the source bits of the associated interrupts. When read, this register returns the source bit values of the associated interrupts. The corresponding interrupts for register SRCSETX are determined through the following formula: SRCSET[31] → (32 × X) + 31 SRCSET[0] → (32 × X) + 0 If ARPROT[1] == 1'b0 (secure) 0 Interrupt is not asserted 1 Interrupt is asserted If ARPROT[1] == 1'b1 (nonsecure) 0 Interrupt is not asserted or configured as secure 1 Interrupt is asserted If AWPROT[1] == 1'b0 (secure) 0 no effect 1 Interrupt source bit set If AWPROT[1] == 1'b1 (nonsecure) 0 no effect 1 Interrupt source bit set if bit is configured as non-secure

57.3.3.11 Source Clear Registers

The Source Clear Registers (SRCCLEAR), shown in Figure 57-12, are used to determine which interrupts are currently asserted and to clear interrupts from the asserted state that were triggered by software, either through the SRCSET or SWINT registers. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, when read, if its source bit is set to 1'b1, the interrupt is asserted and is pending action from the CPU. If its bit is set to 1'b0, the interrupt is not asserted.

Writing a bit to 1'b1 in this register will clear an asserted interrupt that was triggered by software. Writing a bit to 1'b0 will have no effect. To set a source bit, the Source Set Registers or Software Triggered Interrupt Register must be used.

The values read in the SRCCLEAR register represent the currently asserted interrupts.

These registers can only be accessed by 32-bit supervisor accesses.

0xBASE+0x0280 (SRCCLEAR0)
0xBASE+0x0284 (SRCCLEAR1)
0xBASE+0x0288 (SRCCLEAR2)
0xBASE+0x028C (SRCCLEAR3)

Access: Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	SRCCLEAR[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	SRCCLEAR[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-12. Interrupt Source Clear Registers

Table 57-13 shows the SRCCLEAR field descriptions.

Table 57-13. SRCCLEAR Field Descriptions

Field	Description
31–0 SRCCLEAR	Interrupt Source Clear. When written, this register forces the source bits of the associated interrupts. When read, this register returns the source bit values of the associated interrupts. The corresponding interrupts for register SRCCLEARX are determined through the following formula: SRCCLEAR[31] → (32 × X) + 31 SRCCLEAR[0] → (32 × X) + 0 If ARPROT[1] == 1'b0 (secure) 0 Interrupt is not asserted 1 Interrupt is asserted If ARPROT[1] == 1'b1 (nonsecure) 0 Interrupt is not asserted or configured as secure 1 Interrupt is asserted If AWPROT[1] == 1'b0 (secure) 0 no effect 1 Interrupt source bit cleared If AWPROT[1] == 1'b1 (nonsecure) 0 no effect 1 Interrupt source bit cleared if bit is configured as non-secure

57.3.3.12 Priority Registers

The Priority Registers (PRIORITY), shown in Figure 57-13, are used to configure the priority of each interrupt for masking purposes.

Each eight bit priority field is affected by an integration parameter which determines the available resolution of priority levels. The number of bits available can vary from four to eight, with the lowest resolution only implementing bits [7–4], and the highest resolution implementing bits [7–0]. The

implemented number of bits can be found in a manner similar to that described in Section 57.3.3.4, Priority Mask Register."

The priority scheme used is such that an interrupt assigned to 0 has the highest priority.

These registers can only be accessed by 32-bit or 8-bit supervisor transactions.

0xBASE+0x0400 (PRIORITY0)
0xBASE+0x0404 (PRIORITY1)
0xBASE+0x0408 (PRIORITY2)
...
0xBASE+0x0478 (PRIORITY30)
0xBASE+0x047C (PRIORITY31)

Access: Privileged
Read-Write

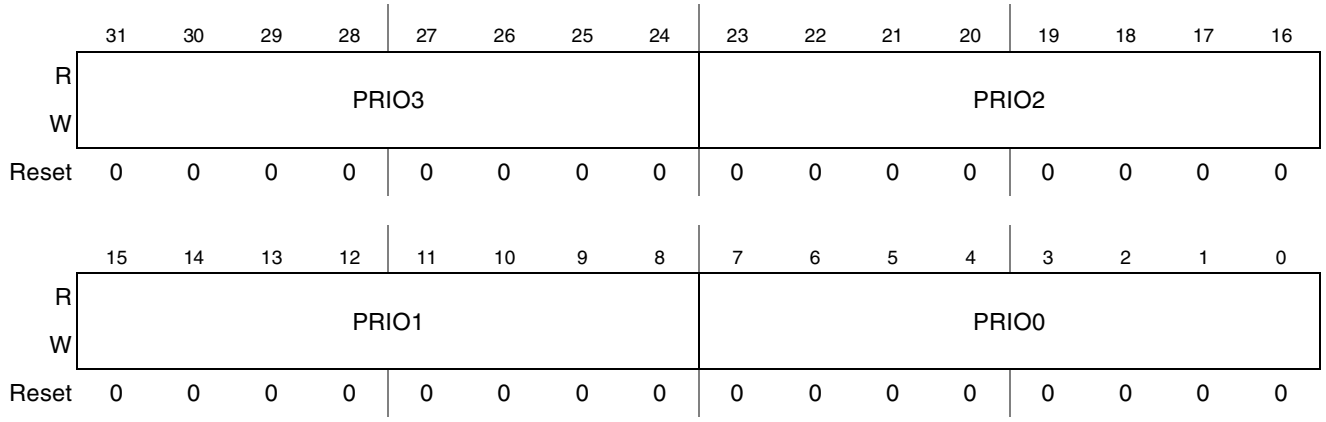


Figure 57-13. Interrupt Priority Registers

Table 57-14 shows the PRIORITY field descriptions.

Table 57-14. PRIORITY Field Descriptions

Field	Description
31-24 PRIO3 23-16 PRIO2 15-8 PRIO1 7-0 PRIO0	Interrupt Priority. This field stores the priority value for its corresponding interrupt. The interrupts for register PRIORITYX are determined through the following formula: PRIO3 → (4 × X) + 3 PRIO2 → (4 × X) + 2 PRIO1 → (4 × X) + 1 PRIO0 → (4 × X) + 0 If ARPROT[1] == 1'b0 (secure) 00000000 Interrupt priority is set to 0 00000001 Interrupt priority is set to 1 ... 11111111 Interrupt priority is set to 255 If ARPROT[1] == 1'b1 (nonsecure) 00000000 Interrupt priority is set to 128 or interrupt is configured as secure 00000010 Interrupt priority is set to 129 ... 11111110 Interrupt priority is set to 255 If AWPROT[1] == 1'b0 (secure) 00000000 Interrupt priority is set to 0 00000001 Interrupt priority is set to 1 ... 11111111 Interrupt priority is set to 255 If AWPROT[1] == 1'b1 (nonsecure) 00000000 Interrupt priority is set to 128 if interrupt is configured as non-secure 00000001 Interrupt priority is set to 128 if interrupt is configured as non-secure 00000010 Interrupt priority is set to 129 if interrupt is configured as non-secure ... 11111110 Interrupt priority is set to 255 if interrupt is configured as non-secure 11111111 Interrupt priority is set to 255 if interrupt is configured as non-secure

57.3.3.13 Pending Registers

The Pending Registers (PND), shown in Figure 57-14, are used to determine which interrupts are currently pending action from the CPU. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, if its pending bit is set to 1'b1, the interrupt is enabled in the ENSET/ENCLEAR registers and asserted either on the interrupt inputs or in the SRCSET/SRCCLEAR registers, and therefore is pending action from the CPU. Priority values have no effect on whether an interrupt is pending.

This is a read-only register, writes will be ignored.

These registers can only be accessed by 32-bit supervisor transactions.

0xBASE+0x0D00 (PND0)
0xBASE+0x0D04 (PND1)
0xBASE+0x0D08 (PND2)
0xBASE+0x0D0C (PND3)

Access: Privileged
Read-Only

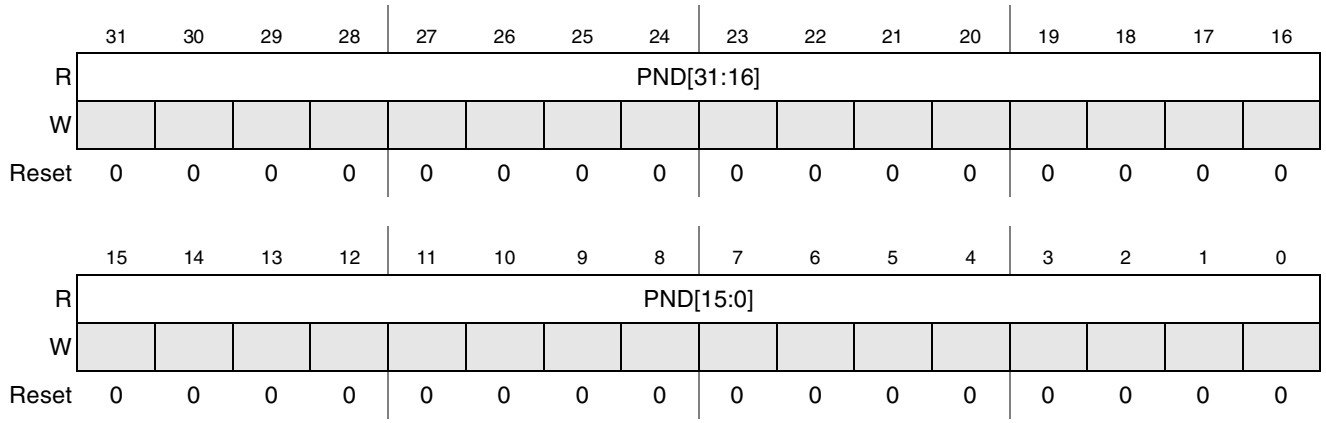


Figure 57-14. Interrupt Pending Registers

Table 57-15 shows the PND field descriptions.

Table 57-15. PND Field Descriptions

Field	Description
31–0 PND	Interrupt Pending Status. When read, this register returns the pending bit values of the associated interrupts, indicating which interrupts are enabled and asserted. The corresponding interrupts for register PNDX are determined through the following formula: PND[31] → (32 × X) + 31 PND[0] → (32 × X) + 0 If ARPROT[1] == 1'b0 (secure) 0 Interrupt is not pending 1 Interrupt is pending If ARPROT[1] == 1'b1 (nonsecure) 0 Interrupt is not pending or configured as secure 1 Interrupt is pending

57.3.3.14 High Priority Pending Registers

The High Priority Pending Registers (HIPND), shown in Figure 57-15, are used to determine which interrupts are currently pending at the highest active priority. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, if its pending bit is set to 1'b1, the interrupt is pending action from the CPU and its priority is the highest level amongst all pending enabled interrupts. See Section 57.3.3.13, Pending Registers" for more information about pending interrupts.

The highest active priority is determined based on the security state of the read transaction. If the register is read by a non-secure transaction, the pending interrupts returned are those non-secure interrupts that are pending and set to the highest priority of any currently pending non-secure interrupts. If the register is read by a secure transaction, the pending interrupts returned are those interrupts, secure or non-secure, that are

pending and set to the highest priority of any currently pending interrupts regardless of security configuration.

This is a read-only register, writes will be ignored.

These registers can only be accessed by 32-bit supervisor transactions.

0xBASE+0x0D80 (HIPND0)
0xBASE+0x0D84 (HIPND1)
0xBASE+0x0D88 (HIPND2)
0xBASE+0x0D8C (HIPND3)

Access: Privileged
Read-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HIPND[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIPND[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-15. High Priority Interrupt Pending Registers

Table 57-16 shows the HIPND field descriptions.

Table 57-16. HIPND Field Descriptions

Field	Description
31–0 HIPND	High Priority Interrupt Pending Status. When read, this register returns the pending bit values of the associated interrupts which are set to the highest currently active priority level. The corresponding interrupts for register HIPNDX are determined through the following formula: HIPND[31] → (32 × X) + 31 HIPND[0] → (32 × X) + 0 If ARPROT[1] == 1'b0 (secure) 0 Interrupt is not pending at the highest priority level 1 Interrupt is pending at the highest priority level If ARPROT[1] == 1'b1 (nonsecure) 0 Interrupt is not pending at the highest priority level or configured as secure 1 Interrupt is pending at the highest priority level

57.3.3.15 Wakeup Configuration Registers

The Wakeup Configuration Registers (WAKEUP), shown in Figure 57-16, are used to set which interrupts will assert the tzic_wakeup_request signal. Each bit in the register corresponds to an interrupt source available in the system. For each interrupt, if its wakeup bit is set to 1'b1 assertion of the interrupt will cause the wakeup request signal to assert combinationally. When its wakeup bit is set to 1'b0, the interrupt will not assert the wakeup request signal.

The values read in the WAKEUP register represent interrupts currently set to wake up the system. Out of reset, all interrupts are configured not to wake up the system. Non-secure transactions can only enable or

disable wakeup configuration of interrupts configured as non-secure. Secure transactions can enable or disable wakeup configuration of any interrupts.

These registers can only be accessed by 32-bit supervisor transactions.

0xBASE+0x0E00 (WAKEUP0)
0xBASE+0x0E04 (WAKEUP1)
0xBASE+0x0E08 (WAKEUP2)
0xBASE+0x0E0C (WAKEUP3)

Access: Privileged
Read-Write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WAKEUP[31:16]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WAKEUP[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-16. Interrupt Wakeup Configuration Registers

Table 57-17 shows the WAKEUP field descriptions.

Table 57-17. WAKEUP Field Descriptions

Field	Description
31–0 WAKEUP	Wakeup Configuration. When read, this register returns the wakeup bit values of the associated interrupts. When written, this register sets the configuration for the associated interrupts. The corresponding interrupts for register WAKEUPX are determined through the following formula: WAKEUP[31] → (32 × X) + 31 WAKEUP[0] → (32 × X) + 0 If AxPROT[1] == 1'b0 0 Interrupt will not assert tzic_wakeup_request 1 Interrupt will assert tzic_wakeup_request If AxPROT[1] == 1'b1 0 Interrupt will not assert tzic_wakeup_request or is configured as secure and not accessible 1 Interrupt will assert tzic_wakeup_request

57.3.3.16 Software Interrupt Register

The interrupt controller software interrupt register (SWINT), shown in Figure 57-17, can be used by software to assert or negate an interrupt at a specific interrupt number. Writing the number of the desired interrupt will either trigger an interrupt to assert if not already asserted by hardware, or negate an interrupt that was originally asserted by software. An interrupt asserted by hardware cannot be negated by this register. Triggering an interrupt with the SWINT is equivalent to asserting the associated interrupt input pin, except regarding the tzic_wakeup_request output. It will be asserted in the SRCSET/SRCCLEAR registers, but will not be pending unless the interrupt is enabled. A software interrupt will not cause a wakeup request.

A secure write can trigger any enabled interrupt. A non-secure write can only trigger non-secure interrupts, an attempt to trigger a secure interrupt with a non-secure write will be ignored. If wbstrb[3] is not asserted to indicate whether asserting or negating an interrupt, the write will be ignored.

This register can only be accessed by 32-bit supervisor transactions.

0xBASE+0x0F00 (SWINT)

Access: Privileged Write-Only

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	INT NEG															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W							INTID									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 57-17. Software Interrupt Trigger Register

Table 57-18 shows the SWTRIG field descriptions.

Table 57-18. SWTRIG Field Descriptions

Field	Description
31 INTNEG	Interrupt Negate. This bit indicates whether the interrupt ID in the INTID field is being asserted or negated. Assuming the interrupt is not being asserted by hardware, when 1'b0, an interrupt is asserted, when 1'b1, an interrupt is negated.
30–10	Reserved
9–0 INTID	Interrupt ID. This field indicates the number of the interrupt to be triggered (asserted or negated). Triggering an interrupt outside of the maximum number of interrupts will result in UPREDICTABLE behavior. Triggering a secure interrupt with a non-secure write will have no effect.

57.4 Functional Description

The TZIC is a trustzone compatible interrupt controller. This implementation supports up to 128 interrupts. It supports autovectoring solutions and provides a high priority pending register for software to directly access the interrupts currently pending at the highest active priority level (based on security configuration).

The TZIC is made up of three primary components:

1. The AXI Interface
2. The Register Block
3. The Interrupt Engine

These three components handle all functionality of the TZIC.

57.4.1 Security Configurability

The TZIC is designed to handle two levels of secure interrupts using trustzone functionality included in the AXI spec. It is also possible to use it in a single security level, non-trustzone aware system. At integration the TZIC can be connected such that all transactions are treated as secure, providing the CPU with full access to all registers. The INTSEC registers reset to all interrupts being secure, which translates to all interrupts being available to the system without needing to be configured, therefore the non-secure software running as secure does not need to be aware of the security registers for configuration.

The INTSEC registers determine which interrupts are accessible by non-secure transactions. Interrupts configured as secure are not accessible or configurable by non-secure transactions. Any information associated with an interrupt configured as secure is not accessible by non-secure transactions. Read transactions will return reset values for any bits tied to a secure interrupt, writes to these bits will be ignored. This includes all registers. The INTSEC registers are not visible to non-secure transactions.

57.4.1.1 TrustZone and Interrupt Priority

The priority scheme implemented in the TZIC allows for flexible prioritization of secure and non-secure interrupts, and ease of use with legacy software.

Each interrupt can be assigned a priority value up to 8 bits (depending on the PRIORITY_MASK parameter). The highest priority value is 0, the next is 1, and so on. For designs where not all 8 bits are used, the lower bits are masked off, such that an implementation with only 5 bits would use values 0 (8’h00), 8 (8’h08), 16 (8’h10), 24 (8’h18), and so on. The number of bits available can be discovered by software through writing 8’hFF to a priority register and reading back the resulting value actually stored in the register.

Secure and non-secure transactions have different views of the priority registers. A secure transaction can see all implemented bits and has access to the full range of priority values. A non-secure transaction is limited to the lower half of the possible priorities. This is accomplished through shifting the values written and read during a non-secure transaction.

Table 57-19. Valid Priority Values Based on Transaction Security

Priority Value	Non-Secure Transaction	Secure Transaction
0 127	No Access	Recommended Values
128 255	Accessible Values	Not Recommended

A non-secure write to a priority register is automatically shifted one bit to the right, with the MSB assigned 1’b1. This shift is reversed when read, with the priority value shifted one bit to the left and the LSB assigned 1’b0. This is shown in the following equations:

Non-Secure write to priority register: priority_reg[7:0] = {1’b1, wdata[7:1]} ***Eqn. 57-1***

Non-Secure read from priority register: rdata[7:0] = {priority_reg[6:0], 1’b0} ***Eqn. 57-2***

This guarantees the highest priority a non-secure transaction can write is 128 (8’h80), which is the highest priority of the lower half of possible priorities. It also shows the non-secure software will not be aware of this limitation, since reading back a priority register with a value of 128 will result in a read data value of 0.

This scheme will allow for non-secure interrupts to function in their own priority scheme without knowledge of secure interrupt priorities while allowing secure interrupts to guarantee priority over non-secure interrupts. If all secure interrupts are assigned priorities in the upper half of the possible priorities, secure interrupts will always be of higher priority than non-secure interrupts.

Note that this priority value shifting is determined by the security of the transaction, not the security of the interrupt whose priority is being configured. A non-secure transaction will only have access to non-secure interrupt priority values, but a secure transaction will have access to secure and non-secure priority values, and will be accessing the actual value, not the shifted value. This means it is possible for a secure transaction to configure a non-secure interrupt priority to be in the upper half of the priority levels, or to configure a secure interrupt priority to be below a non-secure interrupt. In the first case, the non-secure interrupt priority will remain set until written again by another secure transaction or a non-secure transaction (which would automatically limit the priority level to 128 or lower). A non-secure read will shift the value as specified above but would not alter the value. This is not recommended since it could impact the non-secure software in an unpredictable manner. In the second case, it may be desirable to configure a secure interrupt to be of lower priority than some non-secure interrupts, although such behavior should be used with care.

57.4.2 AXI Interface

The AXI interface allows access to all functionality of the TZIC. It is fully compatible with the AMBA 3 AXI specification from ARM.

Read transactions are generally returned with one initial wait state and two clock cycles per read data. All registers can be read with 32-bit transactions, the priority registers allow 8-bit transactions. 16-bit transactions are not supported and will result in an error. Exclusive reads are not supported. The TZIC will accept one read address at a time. If a second read transaction is valid before completion of the previous read, it will be held off until the TZIC is finished returning data from the first read.

Write transactions are generally accepted with one initial wait state, the one clock cycle per write data. Registers are word writeable only with the exception of the priority and priority mask registers which support byte writes. Exclusive writes are not supported. The TZIC will accept one write address at a time. If a second write transaction is valid before completion of the previous write, it will be held off until the TZIC is finished returning the write response from the first write. The TZIC will also hold off the write data bus until it receives a write address. It follows that any interconnect port the TZIC is connected to must be configured for an interleave depth of 1.

57.4.3 Register Block

The Register Block handles all register related behavior for the TZIC. It contains programmable registers for enabling interrupts, setting priority levels, configuring security settings, etc. It also collects current interrupt activity data from the Interrupt Engine and presents it as readable registers.

The number of flops required to handle all the programmable registers is minimized using parameters. The total number of interrupts the controller can handle is controlled by a parameter set by the integrator. Based on that number of interrupts, the Register Block will reduce the number of flops to the minimum needed, with all remaining bits returning 1’b0 when read.

The Register Block provides enable and configuration information to the Interrupt Engine, and receives all information from the Interrupt Engine relating to the SRCSET/SRCCLEAR, PND, and HIPND registers.

57.4.4 Interrupt Engine

The interrupt engine performs all processing on the raw interrupt information. It receives configuration information from the Register Block, interrupts from the system, and based on that information determines when to assert interrupts to the core.

An interrupt is asserted to the core (irq_b or fiq_b) when all criteria are met as follows:

- The EN bit in the INTCTRL register is enabled, secure for fiq_b or non-secure for irq_b.
- An interrupt is asserted, either on the input intin_b or by writing the SRCSET or SWINT registers.
- The interrupt is configured appropriately for the desired interrupt in the INTSEC register, set as secure for fiq_b or non-secure for irq_b.
- The asserted interrupt is enabled in the ENSET/ENCLEAR register.
- The priority of the asserted interrupt is greater than the PRIOMASK register value. Note: this requires the actual priority value to be greater than the PRIOMASK register value, after taking into account shifting for non-secure transactions.

This TZIC implementation can handle up to 128 interrupts. Each interrupt has an associated interrupt ID number 0–127.

The interrupt synchronizers are part of the interrupt engine. A two-flop synchronizer is implemented for each interrupt input. Clock gating is available for the synchronizers and is configured through the Synchronizer Control (SYNCCTRL) Register. For more information, see Section 57.3.3.5, Synchronizer Control.”

The interrupt engine handles the wakeup functionality of the TZIC, processing the dsm_int_holdoff input and asserting tzic_wakeup_request as necessary.

tzic_wakeup_request is an asynchronous, combinational output determined by the asynchronous intin_b input and the WAKEUP registers. It asserts when any interrupt is asserted on the intin_b input while configured for wakeup. It negates when there are no more interrupts asserted that are configured for wakeup. A software interrupt will not assert this output, since a software interrupt would require a clock to be triggered and the tzic_wakeup_request is intended for enabling clocks and/or power during low power states.

dsm_int_holdoff causes the interrupt synchronizers to stop updating. This is for shutting down clocks in the system. While preparing for low power modes, the CPU may be in a position where it expects no more interrupts to be occuring, but has not yet executed a standby-wait-for-interrupt. The CPU must write to the dsm_int_holdoff bit, then read it back to verify that it was successfully set. If not set, an interrupt occurred during the process before the synchronizers could be stopped and the CPU must process the interrupt and write to the dsm_int_holdoff bit again.

57.4.5 Auto-Vectored Interrupt Handling

The TZIC is intended to be used with auto-vectored interrupt service routines. Once interrupts have been enabled using the ENSET/ENCLEAR registers and the INTCTRL register, the interrupt engine is live and will signal an interrupt to the CPU when an enabled interrupt is asserted. Once the CPU interrupt is triggered, a read to the PND or HIPND registers will determine which interrupts are currently pending.

The PND registers will contain information on every interrupt currently pending, whether asserted by the interrupt input or forced by software. This includes interrupts of priority lower than the PRIOMASK value. The PRIOMASK register limits the assertion of the irq_b/fiq_b signal based on priority, it does not affect which interrupts are visible in the pending registers.

The HIPND registers will contain information on which interrupts are currently pending at the highest pending priority, allowing autovectoring software to choose between the highest priority interrupts without having to process such information on its own. As with the PND register, the HIPND register will return pending values even if the highest pending priority is of equal or lower priority than the PRIOMASK value, although in such a case the interrupt signals will not be asserted.

After choosing a pending interrupt to service, writing the priority of the interrupt being serviced to the PRIOMASK register will allow for interrupt preemption by a higher priority interrupt being asserted later.

57.4.6 Integration Options

The following TZIC configuration parameters are available at integration and impact TZIC's functionality.

- NUMBER_OF_INTS

  This parameter determines the number of interrupts the TZIC will handle. This parameter must be a multiple of 32. Software can determine the maximum number of interrupts by reading the HWCONFIG register. Any multiple of 32 less than 1024 is possible, although if configured for 1024, there will only be 1020 actual interrupts available, interrupt ID numbers 1020–1023 are reserved. As integrated, this implementation of the TZIC supports 128 interrupts, and no interrupts are reserved.
- PRIORITY_MASK

  This parameter determines the number of bits used to specify the priority of an interrupt. The minimum allowed is four bits for non-secure interrupts, which requires at least five bits for a dual security configuration. In the minimum dual security configuration, bits [7–3] are usable and bits [2–0] are always 3'b0. As integrated, this implementation of the TZIC has a total of 5 bits. It supports the minimum dual security configuration described above.

57.4.7 Clocks

The TZIC has one clock input, ACLK. The ACLK input must receive a clock signal at all times for interrupt synchronization purposes. Internal clock gating is used to reduce power consumption. If system clocks are to be shut down for power conservation, it is recommended that ACLK receive a clock whenever the CPU is clocked. If the CPU is clocked and the TZIC is not transactions from the CPU to the TZIC could be missed by the TZIC while the CPU thinks it is being received and processed.

57.4.8 Reset

The TZIC is reset by asserting the areset_b input. Resetting the block will zero out almost all registers, the INTSEC registers will reset to all interrupts secure. Coming out of reset, all interrupts are configured as secure, disabled, and set to priority level 0. Also, the INTCTRL register is set to disable all secure and non-secure interrupts.

57.4.9 Endianness

The TZIC data buses are little endian.

57.5 Initialization Information

Initialization procedures will vary based on whether the TZIC is set up in a single vs. dual security level configuration. This description will handle the case where the TZIC is set up to take advantage of both secure and non-secure interrupts, the single security level case is a subset of this scenario.

Note: When writing to any registers affecting the interrupt outputs to the CPU it is recommended the CPU interrupt inputs be masked to prevent interrupts asserting and negating due to register writes.

When the TZIC comes out of reset, all interrupts are disabled and set as secure. It is advised that the INTCTRL register enables are not activated until all other configuration options have been set up. To set up the TZIC for autovectoring functionality:

1. Secure software sets INTSEC registers appropriately.

   This configures all interrupts as either secure or non-secure, allowing the non-secure software to configure the non-secure interrupts. Until these registers are configured, non-secure software will not be able to configure any registers.

   At this point secure and non-secure software are free to configure their associated interrupts with minimal conflict. Non-secure software cannot affect any register bits associated with secure interrupts. It is the responsibility of the secure software not to overwrite any non-secure interrupt configuration bits.
2. Configure ENSET/ENCLEAR.

   Both secure and non-secure software must enable all interrupts needed. At this point interrupt functionality has yet to be enabled in the INTCTRL register so this will not cause interrupts to occur during initialization.

3. Configure PRIORITY.

   Both secure and non-secure software must set PRIORITY registers to appropriate levels, otherwise all interrupts are set to the lowest priority level as determined by the PRIORITY_MASK parameter.

4. Set PRIOMASK.

   The PRIOMASK register must be set to allow interrupts to occur, out of reset it is set to the lowest priority level which masks all priority levels.

5. Set INTCTRL.

   Both secure and non-secure software must enable interrupt functionality in the INTCTRL register. This will enable the TZIC interrupt output to the CPU.

At this point, the TZIC generates interrupt signals to the CPU. The CPU itself must enable interrupts as well. In this configuration the TZIC is enabled for autovectoring software to access the PND or HIPND registers and determine which interrupt to service.

Chapter 58
TV Encoder (TVE)

58.1 Introduction

The TV Encoder (TVE) is designed to provide direct connection between an Application Processor (AP) and a TV set via analog interfaces. The AP may be both a standalone product and a part of a Wireless Baseband Platform (WBP). Integration of the TVE into the AP and the WBP allows minimizing system complexity and cost. The TV Encoder supports different standard-definition (SD) and high-definition (HD) television standards.

The module is based on mixed (digital and analog) signal processing. It includes three main components:

- TV Signal Processor (TVSP) is a digital module which is responsible for modulation, filtering, generation synchronization signals, upsampling, cable detection control, copy protection and other processing tasks.
- Triple Video Digital-to-Analog Converter (TVDAC) is an analog module which is designed to convert up to three digital video signals to a current.
- Cable Detection Circuit (CDC) is an analog module which provides monitoring of TVDAC output signals for Cable Detection (CD).

58.1.1 Features

The main features of the TVE are as follows:

- SD Mode Features
  - Supported TV standards:
    - NTSC
    - PAL B, D, G, H, and I/M/N
    - 480i
    - 576i
  - Supported output formats:
    - Composite video (CVBS) - up to two simultaneous identical outputs are supported
    - S-video (Y/C), composite video and S-video signals may be output simultaneously
    - YPrPb component
    - RGB component
  - Programmable notch/pre-comb filter for CVBS
  - Switchable pedestal
  - Copy Generation Management System (CGMS) support according to

  - EIA-608b
  - IEC 61880-1
  - EIA-J CPR-1204-1
- Wide-Screen Signaling (WSS) support according to ITU-R BT.1119, ETSI EN 300 294
- Closed Caption insertion capability according to EIA-608b
- Macrovision™ 7.1 copy protection
- Output oversampling up to ×16 for elimination of external analog filters

- HD Mode features
  - Supported TV standards:
    - 720p, 60 Hz
    - 720p, 50 Hz
    - 720p, 30 Hz
    - 720p, 25 Hz
    - 720p, 24 Hz
    - 1080i, 60 Hz
    - 1080i, 50 Hz
    - 1035i, 60 Hz (1920 x1035)
    - 1080p, 30 Hz
    - 1080p, 25 Hz
    - 1080p, 24 Hz
  - Supported output formats:
    - YPrPb component
    - RGB component
  - CGMS support according to
    - EIAJ CPR-1204-2 (CGMS)
    - CEA-805-A (CGMS type B)
  - Output oversampling up to x4 for elimination of external analog filters
- Common SD/HD Mode features
  - Flexible timing and gain control mechanism allowing non-standard parameters (“user mode”)
  - Programmable Chroma digital filters
  - Programmable adaptive Luma digital filters for
    - horizontal and vertical sharpening
    - horizontal and vertical noise reduction
    - de-flickering in interlaced modes
  - Programmable YCrCb to RGB color matrix
  - Output resolution—10 bits (single-ended analog output loaded by a double terminated 75-Ohm cable)

— Cable Detection functionality including
  – Separate cable connection monitoring separately for each TVDAC channel both during normal operation and in standby mode
  – Automatic TVE power on when a TV cable has been connected
  – Detection of connection type (composite, S-video, component)
  – Detection of cable short to the ground

Figure 58-1 depicts the integration of the TVE module in the i.MX51.

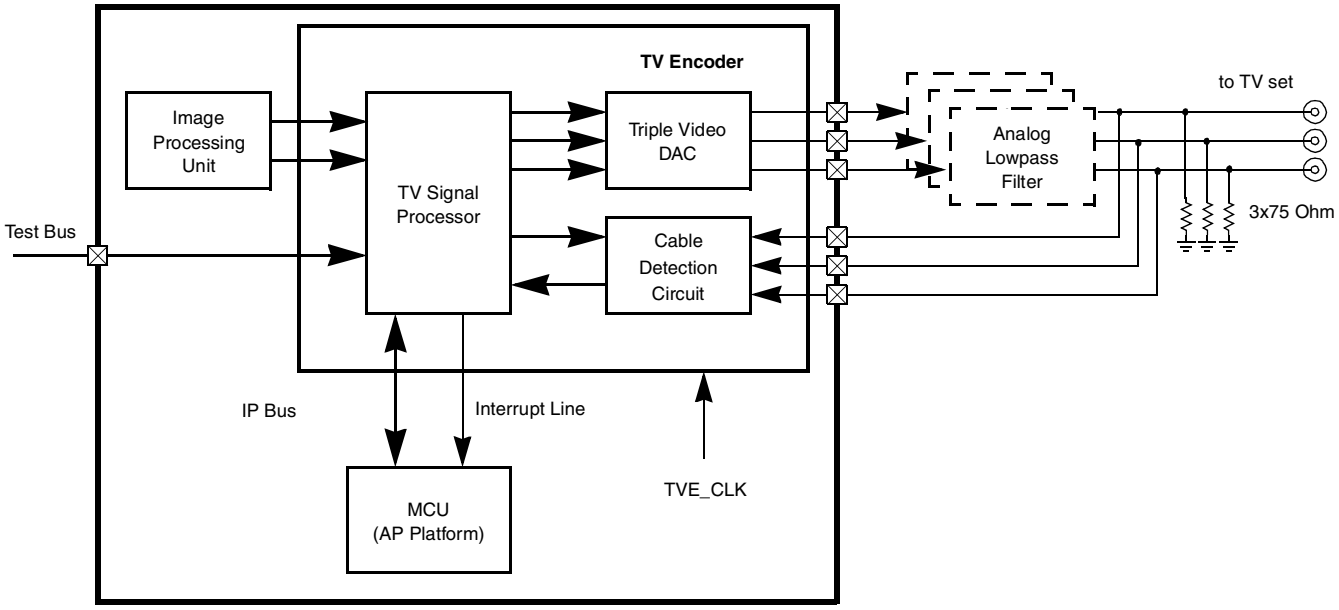


Figure 58-1. i.MX51 TVE Integration Block Diagram

The TVE receives a 30-bit video data stream either via the VIDEO_DATA_IN_1 bus or via the VIDEO_DATA_IN_2 bus from the Image Processing Unit (IPU). In functional mode, only 24-bits are used to transfer the video data in the YCbCr 4:2:2 or YCbCr 4:4:4 formats. Each of YCrCb components is represented by an 8-bit word. In test mode, all 30-bits are used to drive the test data to the TVDAC (10 bits for each TVDAC channel). There is also an option to only drive the 10 least significant bits to all channels when in test mode.

The IPU also provides the sampling clocks VIDEO_DATA_CLK_1 and VIDEO_DATA_CLK_2 for each of the data buses. The clock frequency depends on the mode being used.

In SD mode, the TVE output is always in interlaced format. For some TVE configurations, the TVE performs inter-field vertical filtering for de-flickering, fine sharpening and high frequency noise reduction. In this case, the input data is in progressive format at double the pixel frequency (Fsx). Fsx2 = 27 MHz, where Fs = 13.5 MHz is the pixel frequency for the interlaced format.

In HD mode the input data rate is always Fs = 74.25 MHz or Fs = 74.25/1.001 = 74.1758 MHz. The Y component is sampled at the VIDEO_DATA_CLK_1(2) frequency. For the YCbCr 4:2:2 input format, the

Cb/Cr components are sampled at a half of the VIDEO_DATA_CLK_1(2) frequency. For the YCbCr 4:4:4 input format, the Cb/Cr components are sampled at the VIDEO_DATA_CLK_1(2) frequency.

The stream is also accompanied by two separated signals for horizontal synchronization (HSYNC_1 and HSYNC_2 respectively) and vertical synchronization (VSYNC_1 and VSYNC_2 respectively) and by data valid indicators (VIDEO_DATA_EN_1 and VIDEO_DATA_EN_2 respectively). The timings of the video data stream and the synchronization signals should correspond the ITU-R BT.601 (SD) and SMPTE 296M/274M (HD) requirements.

The second alternative for the TVE input source is the test bus driven by an external device through the chip pins. The test bus can be used for both in functional mode testing or in TVDAC testing. In both cases, the test bus replaces the IPU output.

The TVE generates a TV signal in one of the formats specified above. The signal is oversampled before digital-to-analog conversion. The oversampling factor is programmable and can vary from 2× to 16× in SD mode and from ×2 to ×4 in HD mode. In primary operation mode with maximal output oversampling, the TVDAC sampling frequency is 216 MHz for SD, and 297 MHz or 297/1.001 MHz for HD.

For SD, there are also two additional modes when the TVDAC sampling frequency is 108 or 54 MHz. For HD, there is one additional mode with the TVDAC sampling frequency of 148.5 MHz or 148.5/1.001 MHz. These modes should be used as a reserved option in case the performance of the selected TVDAC degrades at the higher sampling frequency.

The TVDAC is able to directly drive a 75 Ω double terminated cable (the equivalent load is 37.5 Ω). Another option is filtering the TV signal by external Analog Lowpass Filters. The Analog Lowpass Filters may have high (100 Ω) or low (37.5 Ω) input impedance. If 16× oversampling is used, no external analog lowpass filters are required.

The TVE is sampled by the TVE_CLK clock. The clock frequency is equal to the TVDAC sampling rate. All required internal clocks are produced by division of the TVE_CLK inside the TVE. An output voltage of each TVE channel is monitored by Cable Detection (CD) system. The CD is able to distinguish between the following cases:

- Normal load impedance which is equal to 37.5 Ω when the TVE drives directly a 75 Ω double terminated cable
- Double load impedance (75 Ω) when the cable is disconnected
- Zero load impedance when the output is shorted to the ground

The CD monitors each of the output channels separately. The monitoring result is sent to the MCU which determines the TV output mode corresponding to the connected cable type. The TVE is controlled by the ARM platform via the IP interface. The TVE generates interrupts for the ARM platform via a singe interrupt line corresponding to the Indigo IP Interface.

58.1.2 Overview

The TVE block diagram is shown in Figure 58-2. The TVE consists of three modules: the TV Signal Processor (TVSP), the Triple Video DAC (TVDAC) and the Cable Detection Circuit (CDC).

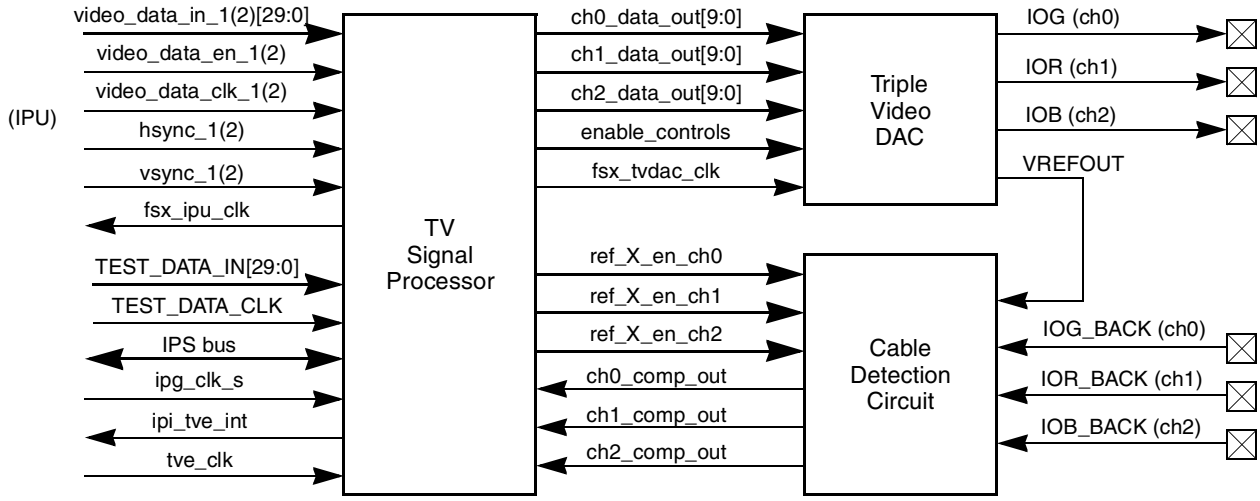


Figure 58-2. TVE Block Diagram

The TVDAC converts an input 10-bit code in the unsigned format to the output current. The DAC is able to directly drive a double terminated 75 Ω cable. The full scale output current is controlled by an external resistor that develops the required full-scale voltage of the video signal on the output load. The nominal sampling rate of the TVDAC is 216 MHz in SD mode and 297 MHz or 297/1.001 MHz in HD mode.

The CDC is an analog module which includes a reference voltage divider and three comparators (one for each TVDAC output). The reference voltage for the CDC is taken from the TVDAC VREFOUT pin. The CDC compares the TVDAC output voltage with the different reference levels derived from the VREFOUT voltage. The result is sent to the TVSP which determines whether or not a cable is connected.

The TVSP is a digital part of the TVE. Figure 58-3 shows the TVSP block diagram.

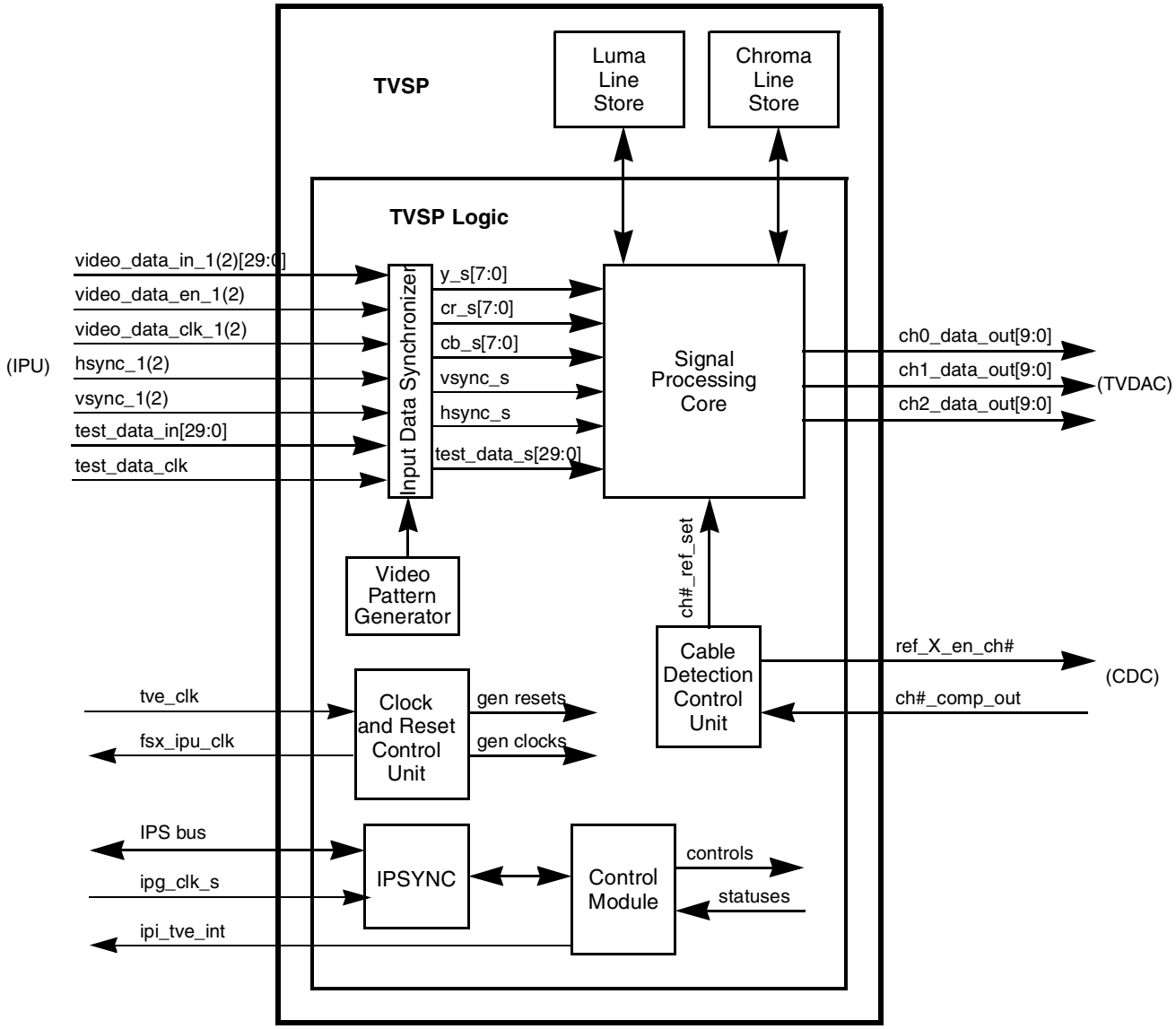


Figure 58-3. TVSP Block Diagram

The TVSP includes two parts: the memory-less TVSP Logic and two memories - the Luma Line Store and the Chroma Line Store.

The central submodule of the TVSP is based on the Signal Processing Core (SPC) which is responsible for the following TV encoding tasks:

- Synchronization signal generation as determined by the TV standard being used
- Digital non-adaptive and adaptive filtering of video signals
- Sampling rate conversion (upsampling)

- Color conversion (YCrCb to RGB) and color adjustment
- Color modulation in composite and S-video output modes as required by the supported SD standards
- VBI data insertion
- Macrovision signal generation for copy protection in SD mode

The SPC receives the input data from the Input Data Synchronizer (IDS) in the YCrCb 4:2:2 or the YCrCb 4:4:4 formats. The IDS is designed to synchronize the IPU output data to the TVE clock. According to the IPU type used, there are two possible synchronization modes.

- If the IPU supports output data synchronization to an external clock, the TVE provides the FSx_IPU_CLK clock (27 MHz in SD mode and 74.25 MHz or 74.25/1.001 MHz in HD mode) to the IPU. The IPU uses this clock to generate the output data stream. The IDS is transparent in this case.
  In functional mode, the sampling rate is 13.5 MHz in SD mode and 74.25 MHz or 74.25/1.001 MHz in HD mode (for the YCrCb 4:2:2 format, the Cb and Cr samples are repeated twice).
  In TVDAC test mode, the data sampling rate is correspondingly 27 MHz and 74.25 MHz or 74.25/1.001 MHz.
- If the IPU does not support output data synchronization to an external clock, it should provide the sampling rates mentioned above (for example, to generate the display clock using fractional frequency division). The frequencies of the IPU output and TVE input clocks must be equal, but a phase jitter between two clocks is allowed. The maximal value of this jitter should be less than one period of one pixel clock and the initial phase difference between two clocks may be unspecified. The IDS is designed to eliminate phase uncertainty and phase jitter between two clocks.

The IDS receives a 30-bit input data from the IPU. In functional mode, the least significant byte of the data bus includes the Y component and the next two bytes are the Cr and Cb components correspondingly. The six most significant bits are unused in functional mode. In TVDAC test mode 1, each ten bits are fed to the corresponding TVDAC channel. In TVDAC test mode 2, only 10 least significant bits are fed to all TVDAC channels and 20 most significant bits are unused.

In addition to the data, the IDS resamples the vertical (VSYNC_1(2)) and horizontal (HSYNC_1(2)) synchronization signals from the IPU. It also gates off the invalid IPU output data according to the VIDEO_DATA_EN_1(2) data enable signal from the IPU.

The SPC has three output channels. Each channel is represented by 10 bits. The outputs can be in composite (CVBS), S-video, YCrCb and RGB component formats according to the TV_OUT_MODE control parameter (see Table 58-1).

Table 58-1. TVEv2 Output Data Formats

Output Channel	Output Data Format (TV_OUT_MODE)							
	Composite (CVBS)				S-video (Y/C)	CVBS and Y/C	YCrCb Component	RGB Component
	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Channel #0	0	CVBS	0	CVBS	Y	Y	Y	G
Channel #1	0	0	0	0	C	C	Cr	R
Channel #2	0	0	CVBS	CVBS	0	CVBS	Cb	B

The Cable Detection Control Unit (CDCU) is a digital component of the CD system. It sends periodic requests to the CDC to monitor the TVDAC output voltage. The CDC returns the results of load resistance measurement and ground short detection for each of the monitored channels. For TVE channels that are inactive for the selected TV_OUT_MODE setting (as indicated by zeros in Table 58-1), the CDCU forces the output current to the value required for the measurement.

The Clock and Reset Control Unit (CRCU) produces internally generated clocks by division of the TVE_CLK clock. The CRCU also asserts reset signals to sub-modules that are unused in the currently selected configuration and enable signals for the TVDAC channels.

The Control Module (CM) is responsible for IP bus interface and interrupt generation support. It contains a set of control and status registers. Because the CM is sampled by the 27 MHz clock in SD mode and by the 74.25-MHz or 74.25/1.001-MHz clock in HD mode, the IP Synchronizer (IPSYNC) is used to synchronize the frequency of the IP bus interface (typically, 66 MHz) to the CM clock frequency.

The SPC block diagram is depicted in Figure 58-4.

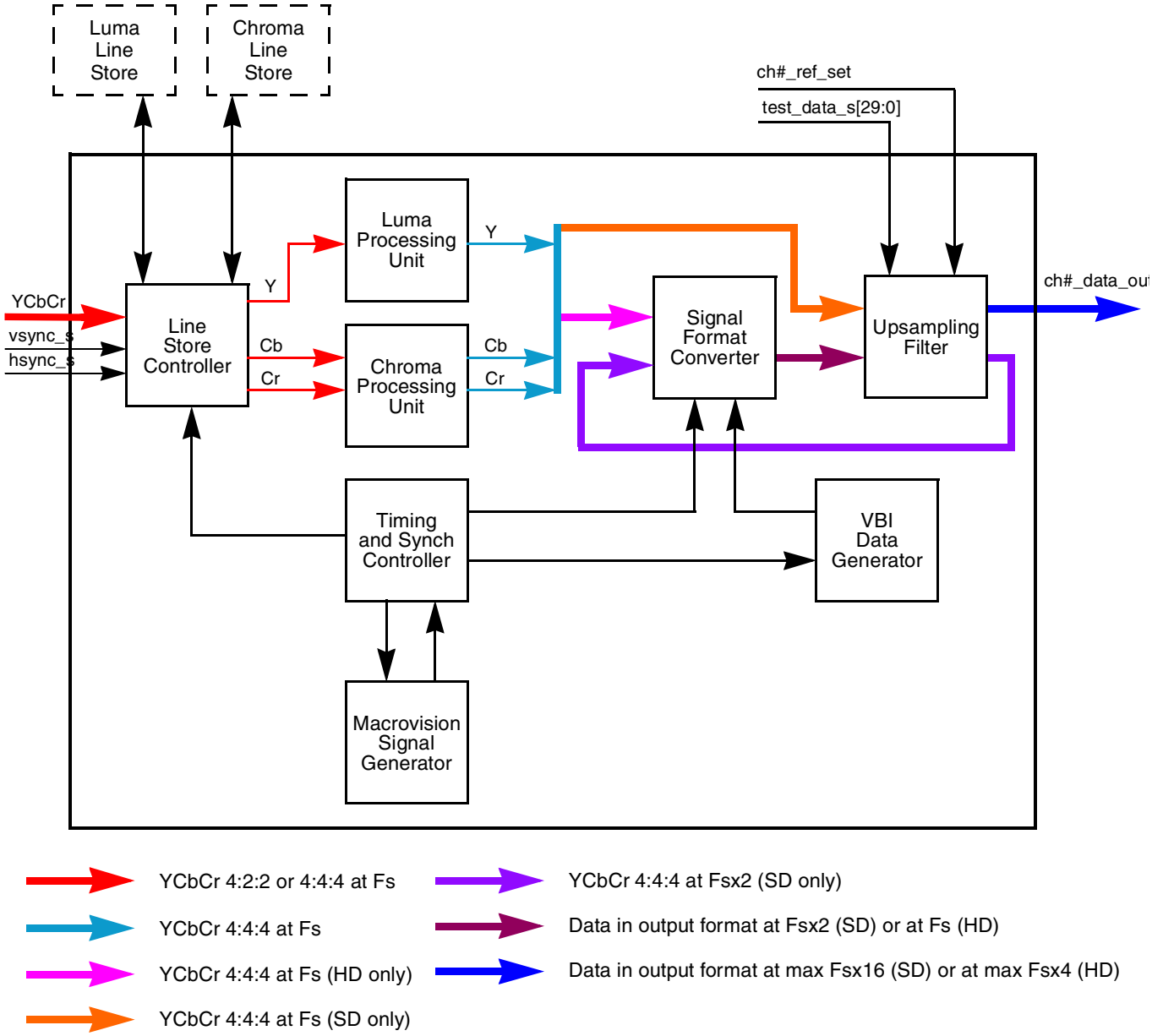


Figure 58-4. SPC Block Diagram

The SPC (shown in Figure 58-4) includes the Line Store Controller (LSC), The Luma Processing Unit (LPU), the Chroma Processing Unit (CPU), the Signal Format Converter (SFC), the Upsampling Filter (UF), The Timing and Synchronization Controller (TSC), the Macrovision signal generator and the VBI Data Generator (VDG).

The LSC receives the data in the YCbCr 4:0:0 or 4:4:4 format from the IDS. The data is written to the LLS (Y) and the CLS (Cb/Cr) line-by-line. The LLS size is 5 lines of 1920 Y samples. The CLS size is 4 lines of 1920 Cb/Cr pairs. The LSC reads 5-sample columns of the Y component from the LLS and sends it to

the LPU. In parallel, the LSC reads 4-sample columns of the Cb/Cr component from the CLS and sends it to the CPU.The columns are read sequentially so for generation of one output line, the whole line of pixel columns is read from the line stores.

The LPU contains a set of filters including:

- 5-tap vertical adaptive filter designed for
  - vertical coarse and fine sharpening
  - vertical mid-frequency and high-frequency noise reduction
  - de-flickering (required only for interlaced TV formats)
- 5-tap horizontal adaptive filter designed for
  - horizontal coarse and fine sharpening
  - horizontal mid-frequency and high-frequency noise reduction
  - horizontal de-ringing
- Notch filter for better Luma/Chroma separation in SD composite mode
- Lowpass filter for reducing out-of-band noise (it may enabled instead of the notch filter because both filters are implemented by the same hardware).

The adaptive filters boost (for sharpening) or suppress (for noise reduction, de-ringing and de-flickering) mid and high frequency components in the Luma spectrum. The filter operation (boost or suppression) depends on the Luma gradient in the corresponding spatial direction. The filters behavior is fully programmable. If the filter for vertical fine sharpening /vertical high-frequency noise reduction/de-flickering is enabled, the input data from the IPU is in the progressive format.

The CPU performs the following operation on the Chroma samples from the CLS:

- Upsampling from Fs/2 to Fs (if the input data format is YCbCr 4:2:2)
- Lowpass filtering for reducing color noise and improving Luma/Chroma separation in SD composite mode
- Comb filtering for improving Luma/Chroma separation in SD composite mode.

The CPU produces the filtered Cb and Cr components sampled at Fs.

The following processing path of the LPU and CPU output samples depends on the mode. In SD mode, the samples are fed to the UF. The UF performs upsampling the data to Fsx2 = 27 MHz. The upsampled data enters the SFC. In HD mode, the LPU and CPU outputs sampled at Fs = 74.25 or 74.25/1.001 MHz are sent directly to the SFC.

The SFC is responsible for the following functions:

- Conversion the YCbCr data to the RGB format for RGB component output
- Subcarrier generation and chroma modulation according to NTSC and PAL standards in SD composite and S-video modes
- Adjustment of brightness, saturation, and hue (in user mode)
- Signal level normalization to the desired values
- Combining the synchronization signals and the VBI data with the video signal
- Composing three output streams according to Table 58-1.

Signal formats and waveforms on three SFC outputs correspond to the final TV signals. The UF provides upsampling of these SFC outputs. The UF consists of three identical channels. Each channel includes the following elements:

- The Gain and Offset Correction (GOC) block is designed to compensate the TVDAC gain and offset deviation from the nominal values. The gain and the offset are adjusted in the range ±25% and ±12.5% of the full scale separately for each channel. The TVDAC_#_GAIN and TVDAC_#_OFFSET parameters are responsible for gain and offset control. The GOC includes an input mux used for selection between the functional mode data, the test data and the CD reference levels.
- The Drop Correction Filter (DCF) provides compensation of the gain drop introduced by the TVDAC at high frequencies. The drop compensation factor is programmable via the TVDAC_#_DROP_COMP parameter separately for each channel.
- The Halfband Upsampling Filter 1 (HUF1) performs upsampling from Fs to Fsx2 (27 MHz in SD mode and 148.5 MHz or 148.5/1.001 MHz in HD mode). In SD mode, the HUF1 input is muxed to the LPU and CPU outputs and the HUF1 output is muxed to the SFC input, so the first upsampling is done before processing in the SFC.
- The Halfband Upsampling Filter 2 (HUF2) performs upsampling from Fsx2 to Fsx4 (54 MHz in SD mode and 297 MHz or 297/1.001 MHz in HD mode). The HUF2 may be bypassed in HD mode if the output sampling frequency is set to Fsx2 via the TVDAC_SAMP_RATE control parameter.
- The 3rd order Interpolation Comb Filter (ICF) performs upsampling from Fsx4 to Fsx8 or Fsx16 in SD mode. The upsampling factor is controlled by the TVDAC_SAMP_RATE parameter. The ICF may be bypassed also in SD mode if the TVDAC_SAMP_RATE sets the output sampling frequency to 54 MHz.

In test mode, a test data is fed directly to the TVDAC (without upsampling) or through the UF.

The TSC is responsible for the following functions:

- Maintenance vertical and horizontal timing specified by the selected TV standard
- Generation of the synchronization signals, the color burst envelope, the active video data enable signal, and the pedestal
- Control of the blanking level for each of three output channels
- Insertion of the Macrovision signal distortions
- Edge shaping for all generated signals
- Delays between the synchronization signals

The TSC is triggered by the VSYNC and HSYNC signals from the IPU. The TSC provides flexible control on horizontal timings in user mode. It also allows non-standard timing modes in addition to the standard configurations (including square pixel operation in SD mode). The MSG controls the insertion of Macrovision signal distortions in the TSC. The MSG receives all necessary triggers from the TSC and returns Macrovision controls to the TSC.

The VDG generates the following VBI control sequences:

- Closed Caption
- CGMS

- WSS

The VDG can be configured only to one of them at the moment because a common hardware is used for generation of all these sequences.

58.1.3 Features

Common TVE features in both SD and HD modes are shown in Table 58-2.

Table 58-2. Common TVE Features in SD and HD Modes

Parameter	Value	Comments
Input data format	YCrCb 4:2:2 YCrCb 4:4:4	The IPU should support 24-bit/pixel parallel data output format
Input interface format and synchronization	Parallel interface with separate HSYNC and VSYNC	Two inputs from the IPU are available to allow transferring video data from one of two IPU display interfaces
Input data resolution	8 bits/component	—
Output data resolution	10 bits	—
Output interface format	Parallel	—
Non-standard horizontal timings support	Yes	Supported in user mode
Programmable adaptive Luma digital filter	Yes. The 5x5 adaptive filter is used for • horizontal and vertical sharpening • horizontal and vertical noise reduction • de-flickering in interlaced modes	The filter is fully programmable. The filter algorithm can be altered in user mode.
Programmable YCrCb to RGB color matrix	Yes	Five matrix coefficients are programmable. The can be also used for brightness, saturation and hue adjustment in user mode.
Cable Detection functionality	Supported. It includes • separate cable connection monitoring separately for each TVDAC channel both during normal operation and in standby mode • automatic TVE power on when a TV cable has been connected • detection of connection type (composite, S-video, component) • detection of cable short to the ground	—

Main parameters of the TVE in SD mode are given in Table 58-3.

Table 58-3. TVE Features in SD Mode

Parameter	Value	Comments
Supported color standards	PAL (B,D,G,H, I/M/N), NTSC	—
Supported TV standards	480i (29.97 Hz), 576i (25 Hz)	—

Table 58-3. TVE Features in SD Mode (continued)

Parameter	Value	Comments
Input frame format	Interlaced if no inter-field filtering is performed. Progressive otherwise.	Inter-field filtering is used for vertical fine sharpening, vertical high-frequency noise reduction and deflickering
Input sampling frequency	If no inter-field filtering is performed: Y - 13.5 MHz Cb/Cr - 6.75 MHz for YCbCr 4:2:2 format, Cb/Cr - 13.5 MHz for YCbCr 4:4:4 format. If inter-field filtering is performed: Y - 27 MHz Cb/Cr - 13.5 MHz for YCbCr 4:2:2 format, Cb/Cr - 27 MHz for YCbCr 4:4:4 format.	—
Square pixel operation	Yes	Also other non-standard timings can be supported in user mode
Output data format	Composite (CVBS) or S-video (Y/C) or component (YPrPb or RGB)	Simultaneous output of two identical CVBS as well as CVBS and Y/C is possible
Output sampling frequency	216/108/54 MHz	Lower sampling frequencies should be used if the TVDAC performance degrades at high sampling frequencies
Macrovision support	Yes. According to Macrovision 7.1	—
Programmable chroma bandwidth	Yes. The possible bandwidth values are: • for YCbCr 4:2:2 input - 0.6, 1.3, 2 and 3 MHz • for YCbCr 4:4:4 input - 1.3, 2.7, 4 and 5.4 MHz	—
Switchable pedestal	Yes	—
Copy generation management system (CGMS) support	Yes. According to EIA-608b, IEC 61880-1, EIA-J CPR-1204-1	—
Closed Caption support	Yes. According to EIA-608b	—
Wide Screen Signaling (WSS) support	Yes. According to ITU-R BT.1119, ETSI EN 300 294	—
Luma notch filter	Yes	For better Luma/Chroma separation in composite mode
Chroma comb filter	Yes	For better Luma/Chroma separation in composite mode

Main parameters of the TVE in HD mode are given in Table 58-4.

Table 58-4. TVE Features in HD Mode

Parameter	Value	Comments
Supported HD TV standards	720p 60 Hz 720p 50 Hz 720p 30 Hz 720p 25 Hz 720p 24 Hz 1080i 60 Hz 1080i 50 Hz 1035i 60 Hz (1920 × 1035) 1080p 30 Hz 1080p 25 Hz 1080p 24 Hz	—
Input frame format	Interlaced for 1080i. Progressive otherwise.	No interfield filtering is available for 1080i
Input sampling frequency	74.25 MHz –for 25/30/60 Hz modes 74.176 MHz - for 29.97/59.94 Hz modes	—
Output data format	Component (YCrCb or RGB)	—
Output sampling frequency	297 MHz –for 25/30/60 Hz modes 296.704 MHz - for 29.97/59.94 Hz modes	—
Programmable chroma bandwidth	Yes. The possible bandwidth values are: • for YCbCr 4:2:2 input—3.3, 7.1, 11 and 16.5 MHz • for YCbCr 4:4:4 input —6.7, 14.3, 22 and 29.7 MHz	—
Copy generation management system (CGMS) support	Yes. According to EIAJ CPR-1204-2 (CGMS) and CEA-805-A (CGMS type B)	—

Main parameters of the TVDAC are given in Table 58-5.

Table 58-5. TVDAC Features

Parameter	Value	Comments
Video signal formats	Composite (CVBS) or S-Video (Y/C) or component (YPrPb or RGB)	—
Resolution	10 bits	—
Signal bandwidth	From 0 to 30 MHz	—
Sampling rate	From 54 to 300 MHz	—
Digital input	Parallel 10x3 bits in unsigned format	—
Output type	Single-end current output	—
Output drive characteristics	Full drive - double-terminated 75 Ohm load (equivalent to 37.5 Ohm) Low drive – 100 Ohm load	Drive with standard output video levels
Output Drive Range	Controllable full-scale current	Controlled by an external resistor
Channel enable control	Separate for each channel and common control enabling the voltage reference	—

Table 58-5. TVDAC Features (continued)

Parameter	Value	Comments
Voltage reference	Build-in bandgap reference	—
Maximal gain mismatch	2%	Between any DAC
Maximal Integral Nonlinearity (INL)	±2 LSB	—
Maximal Differential Nonlinearity (DNL)	±1 LSB	—
Spurious-free Dynamic Range	> 50 dBc at F=1 5 MHz, Fs = 300 MHz	—
Power Supplies	Analog: 2.5–2.75 V Digital: 1.2 V	—

58.1.4 Modes of Operation

This section discusses the following modes of operation:

- Section 58.1.4.1, Normal Operation Mode”
- Section 58.1.4.2, Standby Mode”
- Section 58.1.4.3, Cable Detection Modes”
- Section 58.1.4.4, Test Modes”

58.1.4.1 Normal Operation Mode

In Normal Operation Mode, the TVE converts the input YCrCb 4:2:2 or 4:4:4 video stream to an analog TV signal. The TVE is in Normal Operation Mode, if the TV_OUT_MODE control parameter is in the range from 1 to 7 (see Table 58-1). This parameter also specifies the output TV signal type. During Normal Operation Mode, the inactive TVDAC and UF channels—whose outputs should be 0 according to Table 58-1—are disabled to save power.

The TV standard supported is specified by the TV_STAND control parameter as described in Table 58-6.

Table 58-6. TV_STAND Parameter

TV_STAND	Standard
0000	SD NTSC
0001	SD PALM
0010	SD Combination PALN
0011	SD PAL (B,D,G,H,I)
0100	HD 720p60
0101	HD 720p50
0110	HD 720p30
0111	HD 720p25

Table 58-6. TV_STAND Parameter (continued)

TV_STAND	Standard
1000	HD 720p24
1001	HD 1080i60
1010	HD 1080i50
1011	HD 1035i60 (1920 × 1035)
1100	HD 1080p30
1101	HD 1080p25
1110	HD 1080p24
1111	Reserved

For the TV_STAND values corresponding to HD modes, only the YPbPr and RGB output formats are valid. For SD modes, the input data from the IPU can be in interlaced or progressive formats.

The progressive format is used in the cases when vertical filtering is required for vertical sharpening, noise reduction and de-flickering. The P2I_CONV_EN control bit when set indicates that the input format is progressive. In this case, the IPU should provide progressive frames at the frame rate of 50 fps for PAL and Combinational PALN or at the frame rate of 60/1.001 fps for NTSC and PALM. For all modes, the pixel clock frequency is 27 MHz. The TVE performs vertical filtering of each progressive frame. After that interlacing is done by selection of odd or even lines. Vertical timing relations between input and output video frames for NTSC and PALM are shown in Figure 58-5. Vertical timing relations between input and output video frames for PAL are shown in Figure 58-6. Horizontal timing relations between input and output video lines are shown in Figure 58-7.

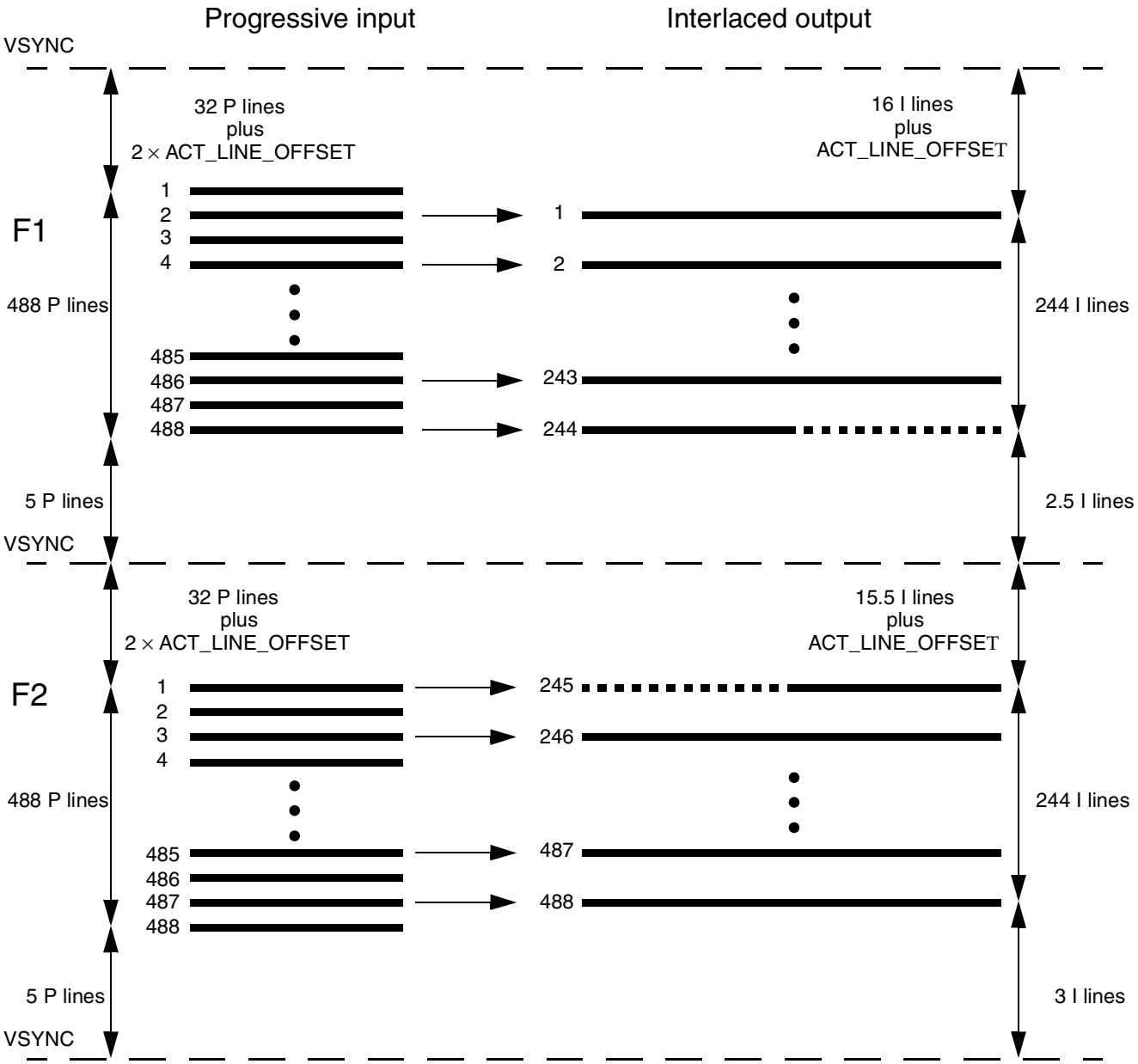


Figure 58-5. NTSC and PALM Vertical Timing Relations for Progressive-to-Interlaced Mode

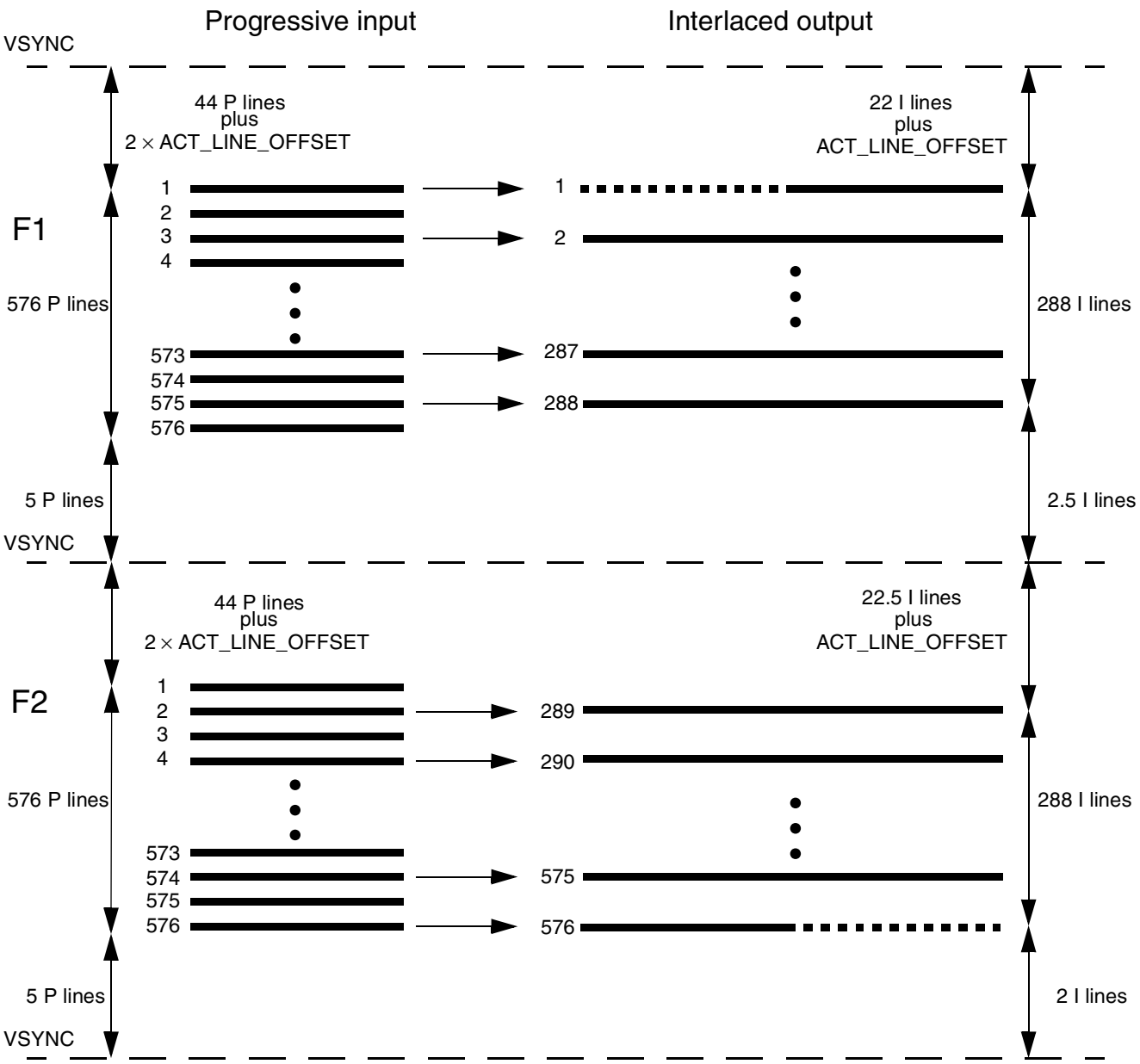


Figure 58-6. PAL Vertical Timing Relations for Progressive-to-Interlaced Mode

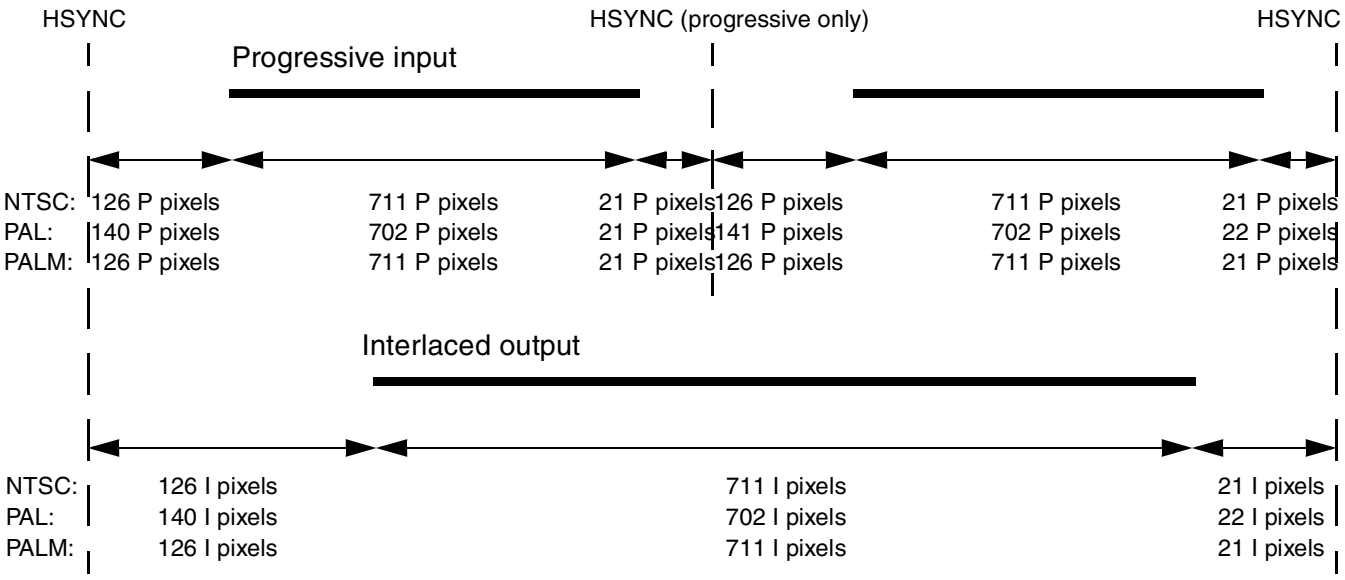


Figure 58-7. Horizontal Timing Relations for Progressive-to-Interlaced Mode

Vertical and horizontal blanking intervals shown in Figure 58-5–Figure 58-7 are minimal. If the IPU is configured to smaller blanking intervals, the TVE cuts a part of image which is outside the minimal blanking intervals. If the IPU is configured to larger blanking intervals, the TVE pads the missing image part between the actual and minimal blanking intervals by black pixels. For progressive-to-interlaced mode, the horizontal blanking interval (the timing interval between the HSYNC and the start of active data) must be an even number of pixels. The minimal vertical blanking interval between the VSYNC and the first active data line may be increased via the ACT_LINE_OFFSET control parameter. This feature may be used for disabling both the data and the pedestal on ACT_LINE_OFFSET first lines in NTSC mode.

If progressive-to-interlaced conversion is disabled, the IPU should provide interlaced vertical timings according the right parts of Figure 58-5 and Figure 58-6 and interlaced horizontal timings according to the bottom part of Figure 58-7. In cases when the IPU blanking intervals differ from those shown in these figures, the TVE cuts the image or pads it with black pixels as described above.

58.1.4.2 Standby Mode

The TVE is in Standby Mode, if the TV_OUT_MODE control parameter is 0. In this case, the most of internally generated clocks are gated off and internal resets are generated for the TVE sub-modules. The TVDAC is disabled. If the CD is enabled, the UF, the CDC and the TVDAC are periodically turned on for a short time to monitor cable connection.

58.1.4.3 Cable Detection Modes

The CD operation is supported with both hardware and software involvement. Different types of the TV connection are distinguished by a number of cable lines connected to the TV output. If one or several cable lines are connected, the equivalent load impedance for these outputs is close to 37.5 Ω and the voltage on

them matches the nominal video levels. If no cable is connected, the load impedance is elevated to 75 Ω. In this case, the output voltage is higher than the nominal video levels.

An additional feature of the CD system is detection of output line short to the ground. The short may appear if a headphone jack is plugged into the A/V connector and shorts the CVBS output to the ground.

Some natural limitations exist for CD operation. It is impossible to differentiate the mixed CVBS/S-video, the component YPrPb and the component RGB connections because in all these three cases a three-wire cable is used. So for these cases the software should set a default option or allow user choosing the interface type.

The CD is supported in both full-drive output mode (when the TV cable is connected directly to the TVDAC outputs) and low-drive output mode (when the TVDAC drives a high-resistance load and a buffer/filter is located between the TVDAC output and the TV connector). In both cases, three monitoring input pins should be connected on a board directly to the cable plug.

There are three CD modes:

- Monitoring the current cable connection when the TVE is in Normal Operation Mode. The mode is enabled if TV_OUT_MODE is not 000 and the CD_TRIG_MODE is 0.
- Periodical automatic pending the cable connection when the TVE is in Standby Mode. The mode is enabled if TV_OUT_MODE is 000 and the CD_TRIG_MODE is 0.
- Checking the cable connection ‘on demand’ (on the trigger control generated by software). The mode is enabled if the CD_TRIG_MODE is 1.

In modes 1 and 2, the CDCU periodically initiates monitoring the output voltage. In mode 3, the MCU should set the CD_MAN_TRIG bit.

The CDC measures the voltage in two steps. At the first step, cable connection is checked. The voltage comparison results obtained at the first step are compared with the current TV output mode specified by the TV_OUT_MODE parameter. If the results do not match the current output mode, an interrupt to the MCU is generated. At the second step, output short to ground is checked. If short has been detected, an interrupt to the MCU is generated.

The software is able to control for which output channels cable connection and/or output short should be checked. The CD_CH_#_LM_EN and CD_CH_#_SM_EN control bits are used for this. If an output load for one of the channels for which the CD_CH_#_LM_EN=1 is different from the value defined the TV_OUT_MODE, the load monitoring interrupt status bit (CD_LM_INT) is set and the corresponding interrupt is generated. If the output of one of the channels for which the CD_CH_#_SM_EN=1 is shorted to the ground, the short monitoring interrupt status bit (CD_SM_INT) is set and the corresponding interrupt is generated.

In the first CD mode, the monitoring period is a multiple of a TV field duration (for SD mode) or a TV frame duration (for HD mode). It is specified by the CD_MON_PER control parameter. The measurement is performed during the first half-line of the pre-equalization pulse in the vertical blanking period.

In the second CD mode, the monitoring period is a multiple of 0.31 sec. It is also specified by the CD_MON_PER control parameter. For output monitoring, the CDCU enables the monitored channels for a measurement time and forces its input to the desired value.

The reference voltage level used for comparison in the CDC can be equal to the Luma blanking level (about 320 mV) or to the Chroma blanking level (about 670 mV). The specific values of the reference levels depend on the CD_REF_MODE and the CD_CH_2_REF_LVL and the current TV output mode specified by the TV_OUT_MODE parameter.

Table 58-7 shows the CD reference levels depending on these control parameters.

Table 58-7. CD Reference Level

Output channel	CD_REF_MODE	CD Reference Level[1] (TV_OUT_MODE)							
		Composite (CVBS)				S-video (Y/C)	CVBS and Y/C	YCrCb component	RGB component
		(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
Channel #0	0	CD_CH_0_REF_LVL	0	CD_CH_0_REF_LVL	0	0	0	0	0
	1	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL	CD_CH_0_REF_LVL
Channel #1	0	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	1	1	CD_CH_1_REF_LVL	0
	1	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL	CD_CH_1_REF_LVL
Channel #2	0	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	0	0	CD_CH_2_REF_LVL	0	CD_CH_2_REF_LVL	0
	1	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL	CD_CH_2_REF_LVL

[1] Luma Reference Level is 0, Chroma Reference Level is 1

58.1.4.4 Test Modes

There are two groups of TVE test modes: the whole TVE is tested or just the TVDAC is tested.

To invoke the test mode wherein the whole TVE is tested, the TVDAC_TEST_MODE control parameter should be 0. The TVE input data, VSYNC and HSYNC are produced by an internal colorbar generator located in the VPG or received from the external test bus. Selection between these two options is done via the DATA_SOURCE_SEL control parameter. The internal generator produces the colorbar according to the selected TVE operational mode defined by the TV_STAND, P2I_CONV_EN and INP_VIDEO_FORM control parameters. The rest of control parameters should be configured as for regular functional mode.

To involve the TVDAC test mode, the TVDAC_TEST_MODE should be set from 1 to 5. TVDAC test modes are designed for TVDAC characterization and production testing. For these modes, the most part of the TVSP processing section is bypassed. The 30-bit input test data may be received either from one of two IPU buses or from the external test bus. The data sampling ate should be 27 MHz for testing the TVDAC in SD mode or 74.25 MHz for testing the TVDAC in HD mode. The IDS output is fed directly to the UF. There are five modification of TVDAC Test Mode selected by the TVDAC_TEST_MODE. In

TVDAC Test Modes 1 and 3, the 30-bit TVE input data is separated into three 10-bit words. Each word is fed via the UF to the corresponding TVDAC channel. In TVDAC Test Modes 2 and 4, only 10 least significant bits of the input data word are sent to all UF and TVDAC channels in parallel. In TVDAC Test Modes 1 and 2, the UF is used to interpolate the test data before driving it to the TVDAC. In TVDAC Test Modes 3 and 4, the UF is bypassed. In TVDAC Test Mode 5, the VPG drives a sine wave signal with programmable frequency and amplitude to the TVDAC inputs. Independently on the TVDAC_TEST_MODE setting, the input of each TVDAC channel can be forced to a programmable constant code defined via the BLANKING_CH_#_USR control parameters. Forcing is done when the corresponding TVDAC_#_DATA_FORCE bit is set.

58.2 External Signal Description

The MMT pins are described in Table 58-8.

Table 58-8. Detailed Signal Descriptions

Signal	I/O	Description	
Clocks and resets			
tve_clk	I	High speed clock input	
		State Meaning	Asserted/Negated—Samples TVE flip-flops on positive edge
		Timing	Assertion/Negation - According to the required operation frequency. The allowed frequencies for SD are 216, 108, 54 MHz, for HD - 297 (297/1.001), 148.5 (148.5/1.001) MHz
ipg_clk_s	I	IPI Green Line gated clock for IP bus	
		State Meaning	According to the Green Line SRS protocol
		Timing	According to the Green Line SRS protocol. May be asynchronous with the tve_clk clock
fsx_ipu_clk	O	Internally generated clock fed to the IPU	
		State Meaning	Used for sampling of the output data interface in the IPU
		Timing	Derived by division of the TVE_CLK clock. Edge-synchronized with TVE_CLK
ipg_hard_async_reset_b	I	System HW reset	
		State Meaning	According to the Green Line SRS protocol
		Timing	According to the Green Line SRS protocol

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
ipg_soft_reset_b	I	System SW reset	
		State Meaning	According to the Green Line SRS protocol
		Timing	According to the Green Line SRS protocol
Input Data Interface			
video_data_clk_1	I	Clocking signal for input video data bus 1 from the IPU	
		State Meaning	Negative edge — Indicates that the IPU has updated the video_data_in_1 bus Positive edge — Indicates that the TVE can sample the video_data_in_1 bus
		Timing	The clock frequency is 13.5 MHz or 27 MHz in functional SD mode, 74.25 (74.25/1.001) MHz in functional HD mode. The clock frequency is 27 MHz (SD) and 74.25 (74.25/1.001) MHz (HD) in TVDAC test modes. The clock may be asynchronous with the TVE_CLK but the phase jitter between this clock and the clock generated in the TVE must be less than a half of video_data_clk cycle.
video_data_in_1[29:0]	I	Video data input bus 1 from the IPU	
		State Meaning	In functional mode - video_data_in_1[7:0]: Y video_data_in_1[15:8]: Cr video_data_in_1[23:16]: Cb video_data_in_1[29:24]: unused In TVDAC test modes 1 and 3 - video_data_in_1[9:0]: test data for TVDAC channel #0 video_data_in_1[19:10]: test data for TVDAC channel #1 video_data_in_1[29:20]: test data for TVDAC channel #2 In TVDAC test modes 2 and 4 - video_data_in_1[9:0]: test data for TVDAC channels #0 - #2 video_data_in_1[29:10]: unused
		Timing	The bus is sampled by negative edges of the video_data_clk_1 clock
video_data_en_1	I	Enable signal for input video data bus 1 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates valid input video data on the bus 1 Negated— Indicates no valid input video data on the bus 1
		Timing	Assertion/Negation — Synchronized to negative edges of the video_data_clk_1 clock.

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
vsync_1	I	Vertical synchronization signal for input video data bus 1 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates video frame start Negated— Indicates no video frame start
		Timing	Assertion/Negation— Synchronized to negative edges of the video_data_clk_1 clock. Remains asserted for one clock cycle
hsync_1	I	Horizontal synchronization signal for input video data bus 1 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates video line start Negated— Indicates no video line start
		Timing	Assertion/Negation — Synchronized to negative edges of the video_data_clk_1 clock. Remains asserted for one clock cycle
video_data_clk_2	I	Clocking signal for input video data bus 2 from the IPU	
		State Meaning	Negative edge — Indicates that the IPU has updated the video_data_in_2 bus Positive edge — Indicates that the TVE can sample the video_data_in_2 bus
		Timing	The clock frequency is 13.5 MHz or 27 MHz in functional SD mode, 74.25 (74.25/1.001) MHz in functional HD mode. The clock frequency is 27 MHz (SD) and 74.25 (74.25/1.001) MHz (HD) in TVDAC test modes. The clock may be asynchronous with the TVE_CLK but the phase jitter between this clock and the clock generated in the TVE must be less than a half of video_data_clk cycle.
video_data_in_2[29:0]	I	Video data input bus 2 from the IPU	
		State Meaning	In functional mode - video_data_in_2[7:0]: Y video_data_in_2[15:8]: Cr video_data_in_2[23:16]: Cb video_data_in_2[29:24]: unused In TVDAC test modes 1 and 3 - video_data_in_2[9:0]: test data for TVDAC channel #0 video_data_in_2[19:10]: test data for TVDAC channel #1 video_data_in_2[29:20]: test data for TVDAC channel #2 In TVDAC test modes 2 and 4 - video_data_in_2[9:0]: test data for TVDAC channels #0 - #2 video_data_in_2[29:10]: unused
		Timing	The bus is sampled by negative edges of the video_data_clk_2 clock

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
video_data_en_2	I	Enable signal for input video data bus 2 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates valid input video data on the bus 2 Negated— Indicates no valid input video data on the bus 2
		Timing	Assertion/Negation — Synchronized to negative edges of the video_data_clk_2 clock.
vsync_2	I	Vertical synchronization signal for input video data bus 2 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates video frame start Negated— Indicates no video frame start
		Timing	Assertion/Negation— Synchronized to negative edges of the video_data_clk_2 clock. Remains asserted for one clock cycle
hsync_2	I	Horizontal synchronization signal for input video data bus 2 from the IPU. In TVDAC test modes, the signal is unused.	
		State Meaning	Asserted—Indicates video line start Negated— Indicates no video line start
		Timing	Assertion/Negation — Synchronized to negative edges of the video_data_clk_2 clock. Remains asserted for one clock cycle
External Test Data Interface			
IPP_IND_TEST_DATA_CLK	I	Clocking signal for input data from the external test bus	
		State Meaning	Negative edge — Indicates that an external source has updated the IPP_IND_TEST_DATA_IN bus Positive edge — Indicates that the TVE can sample the IPP_IND_TEST_DATA_IN bus
		Timing	The clock frequency is 13.5 or 27 MHz (SD) and 74.25 (74.25/1.001) MHz (HD) in TVE test mode. The clock frequency is 27-MHz (SD) and 74.25 (74.25/1.001) MHz (HD) in TVDAC test modes. The clock may be asynchronous with the TVE_CLK but the phase jitter between this clock and the clock generated in the TVE must be less than a half the clock cycle.

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPP_IND_TEST_DATA_IN[29:0]	I	Test data input from an external source	
		State Meaning	In TVE test mode - IPP_IND_TEST_DATA_IN[7:0]: Y IPP_IND_TEST_DATA_IN[15:8]: Cr IPP_IND_TEST_DATA_IN[23:16]: Cb IPP_IND_TEST_DATA_IN[24]: data enable IPP_IND_TEST_DATA_IN[25]: hsync IPP_IND_TEST_DATA_IN[26]: vsync IPP_IND_TEST_DATA_IN[29:27]: unused In TVDAC test modes 1 and 3 - IPP_IND_TEST_DATA_IN[9:0]: test data for TVDAC channel #0 IPP_IND_TEST_DATA_IN[19:10]: test data for TVDAC channel #1 IPP_IND_TEST_DATA_IN[29:20]: test data for TVDAC channel #2 In TVDAC test modes 2 and 4 - IPP_IND_TEST_DATA_IN[9:0]: test data for TVDAC channels #0 - #2 IPP_IND_TEST_DATA_IN[29:10]: unused
		Timing	The bus is sampled by negative edges of the IPP_IND_TEST_DATA_CLK clock
IPP_IBE_TEST_DATA_IN	O	Test data bus and clock pins direction control	
		State Meaning	Asserted - Indicates that the test data bus and clock pins direction should be set for test data input from an external source (the external test data is used) Negated - Indicates that the test data bus and clock pins direction should be set according to alternate functions of these pins (the external test data is unused)
		Timing	The signal is sampled by positive edges of the TVE clock
Output Analog Interface and Other Analog Pins			
IPP_ANALOGIO_TVCDC_IOG_BACK	I	Cable detection channel #0 (green) input	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVCDC_IOR_BACK	I	Cable detection channel #1 (red) input	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVCDC_IOB_BACK	I	Cable detection channel #2 (blue) input	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPP_ANALOGIO_TVDAC_VREFIN	I	Reference voltage input	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVDAC_IOG	O	TVDAC channel #0 (green) output	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVDAC_IOR	O	TVDAC channel #1 (red) output	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVDAC_IOB	O	TVDAC channel #2 (blue) output	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVDAC_VREFOUT	O	Reference voltage output	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_ANALOGIO_TVDAC_COMP	na	Reference voltage compensation	
		State Meaning	Voltage range is according to TVDAC specifications
		Timing	na
IPP_POWER_TVDAC_DHVDD	na	TVDAC 2.75 V digital power supply	
		State Meaning	na
		Timing	na

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPP_POWER_TVDAC_DHVSS	na	TVDAC 2.75 V digital ground	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AHVDD	na	TVDAC 2.75 V analog power supply	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AHVSS	na	TVDAC analog ground	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AVDDG	na	TVDAC 2.75 V analog power supply for output channel #0 (green)	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AVDDR	na	TVDAC 2.75 V analog power supply for output channel #1 (red)	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AVDDB	na	TVDAC 2.75 V analog power supply for output channel #2 (blue)	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AVSSG	na	TVDAC analog ground for output channel #0 (green)	
		State Meaning	na
		Timing	na

Table 58-8. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPP_POWER_TVDAC_AVSSR	na	TVDAC analog ground for output channel #1 (red)	
		State Meaning	na
		Timing	na
IPP_POWER_TVDAC_AVSSB	na	TVDAC analog ground for output channel #2 (blue)	
		State Meaning	na
		Timing	na
IPP_POWER_TVCDC_HVDD	na	Cable detection circuit 2.75 V analog power supply	
		State Meaning	na
		Timing	na
IPP_POWER_TVCDC_HVSS	na	Cable detection circuit analog ground	
		State Meaning	na
		Timing	na
Miscellaneous Signals			
via_tvdac_en_pol	I	TVDAC global and channels enable signals polarity	
		State Meaning	Asserted—Indicates that the TVDAC enabling signals are active high. Negated—Indicates that the TVDAC enabling signals are active low.
		Timing	Assertion/Negation—Static signal assigned by TVE integration into SOC.

58.2.1 Interface Between the IPU and the TVE

The timing diagrams of the interface signals between the IPU and the TVE are shown in Figure 58-8 and Figure 58-9.

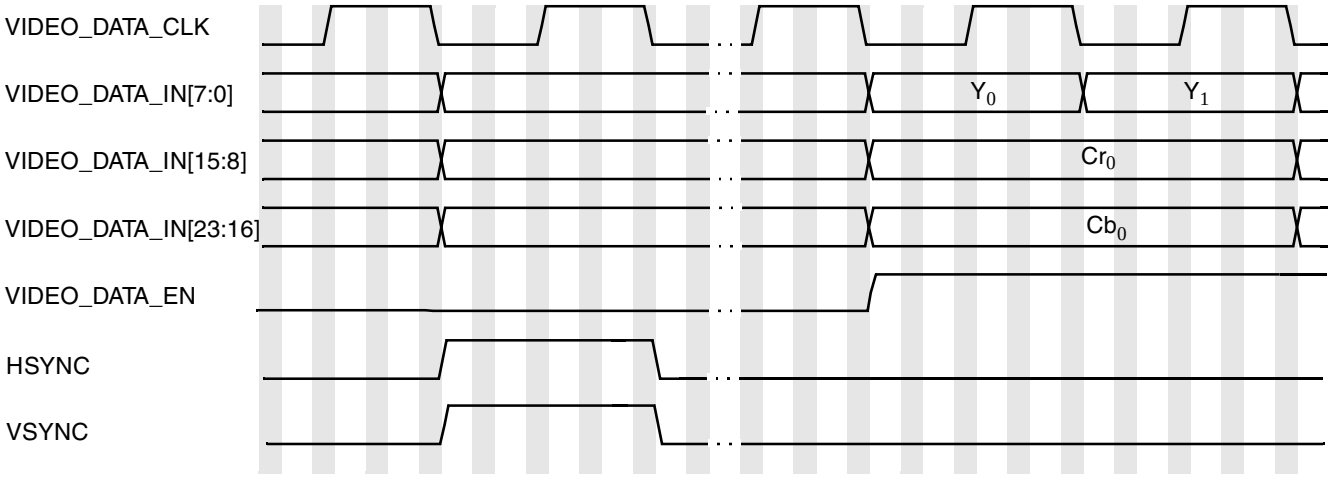


Figure 58-8. Interface between the IPU and the TVE for YCbCr 4:2:2 Input Format

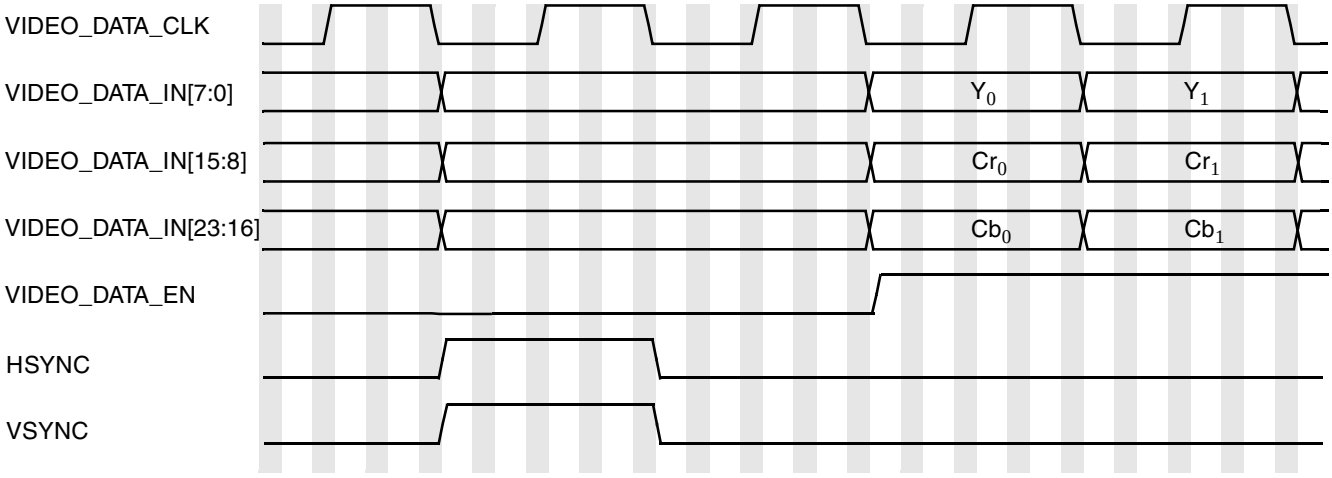


Figure 58-9. Interface between the IPU and the TVE for YCbCr 4:4:4 Input Format

58.3 Memory Map and Register Definition

This section contains a register map, register key, table of register conventions, and register summary.

58.3.1 Register Map

This register map is depicted in Table 58-9.

Table 58-9. TVE Registers

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x0000 (COM_CONF_REG)	Common Configuration Register (COM_CONF_REG)	R/W	0x8004_2000	58.3.3.1/58-43
0xBASE+0x0004 (LUMA_FILT_CONT_REG_0)	Luma Filter Control Register 0 (LUMA_FILT_CONT_REG_0)	R/W	0x0000_0000	58.3.3.2/58-46
0xBASE+0x0008 (LUMA_FILT_CONT_REG_1)	Luma Filter Control Register 1 (LUMA_FILT_CONT_REG_1)	R/W	0x0000_0000	58.3.3.3/58-47
0xBASE+0x000c (LUMA_FILT_CONT_REG_2)	Luma Filter Control Register 2 (LUMA_FILT_CONT_REG_2)	R/W	0x0000_0000	58.3.3.4/58-49
0xBASE+0x0010 (LUMA_FILT_CONT_REG_3)	Luma Filter Control Register 3 (LUMA_FILT_CONT_REG_3)	R/W	0x0000_0000	58.3.3.5/58-50
0xBASE+0x0014 (LUMA_SA_CONT_REG_0)	Luma Statistic Analysis Control Register 0 (LUMA_SA_CONT_REG_0)	R/W	0x0000_0000	58.3.3.6/58-52
0xBASE+0x0018 (LUMA_SA_CONT_REG_1)	Luma Statistic Analysis Control Register 1 (LUMA_SA_CONT_REG_1)	R/W	0x0000_0000	58.3.3.7/58-53
0xBASE+0x001C (LUMA_SA_STAT_REG_0)	Luma Statistic Analysis Status Register 0 (LUMA_SA_STAT_REG_0)	R/W	0x0000_0000	58.3.3.8/58-55
0xBASE+0x0020 (LUMA_SA_STAT_REG_1)	Luma Statistic Analysis Status Register 1 (LUMA_SA_STAT_REG_1)	R/W	0x0000_0000	58.3.3.9/58-56
0xBASE+0x0024 (CHROMA_CONT_REG)	Chroma Control Register (CHROMA_CONT_REG)	R/W	0x0000_0000	58.3.3.10/58-57
0xBASE+0x0028 (TVDAC_0_CONT_REG)	TVDAC 0 Control Register (TVDAC_0_CONT_REG)	R/W	0x0000_0000	58.3.3.11/58-59
0xBASE+0x002C (TVDAC_1_CONT_REG)	TVDAC 1 Control Register (TVDAC_1_CONT_REG)	R/W	0x0000_0000	58.3.3.12/58-61

Table 58-9. TVE Registers (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x0030 (TVDAC_2_CONT_REG)	TVDAC 2 Control Register (TVDAC_2_CONT_REG)	R/W	0x0000_0000	58.3.3.13/58-62
0xBASE+0x0034 (CD_CONT_REG)	Cable Detection Control Register (CD_CONT_REG)	R/W	0x0000_0000	58.3.3.14/58-64
0xBASE+0x0038 (VBI_DATA_CONT_REG)	VBI Data Control Register (VBI_DATA_CONT_REG)	R/W	0x0000_0000	58.3.3.15/58-66
0xBASE+0x003C (VBI_DATA_REG_0)	VBI Data Register 0 (VBI_DATA_REG_0)	R/W	0x0000_0000	58.3.3.16/58-68
0xBASE+0x0040 (VBI_DATA_REG_1)	VBI Data Register 1 (VBI_DATA_REG_1)	R/W	0x0000_0000	58.3.3.17/58-69
0xBASE+0x0044 (VBI_DATA_REG_2)	VBI Data Register 2 (VBI_DATA_REG_2)	R/W	0x0000_0000	58.3.3.18/58-70
0xBASE+0x0048 (VBI_DATA_REG_3)	VBI Data Register 3 (VBI_DATA_REG_3)	R/W	0x0000_0000	58.3.3.19/58-71
0xBASE+0x004C (VBI_DATA_REG_4)	VBI Data Register 4 (VBI_DATA_REG_4)	R/W	0x0000_0000	58.3.3.20/58-72
0xBASE+0x0050 (VBI_DATA_REG_5)	VBI Data Register 5 (VBI_DATA_REG_5)	R/W	0x0000_0000	58.3.3.21/58-73
0xBASE+0x0054 (VBI_DATA_REG_6)	VBI Data Register 6 (VBI_DATA_REG_6)	R/W	0x0000_0000	58.3.3.22/58-74
0xBASE+0x0058 (VBI_DATA_REG_7)	VBI Data Register 7 (VBI_DATA_REG_7)	R/W	0x0000_0000	58.3.3.23/58-75
0xBASE+0x005C (VBI_DATA_REG_8)	VBI Data Register 8 (VBI_DATA_REG_8)	R/W	0x0000_0000	58.3.3.24/58-76
0xBASE+0x0060 (VBI_DATA_REG_9)	VBI Data Register 9 (VBI_DATA_REG_9)	R/W	0x0000_0000	58.3.3.25/58-77
0xBASE+0x0064 (INT_CONT_REG)	Interrupt Control Register (INT_CONT_REG)	R/W	0x0000_0000	58.3.3.26/58-78
0xBASE+0x0068 (STAT_REG)	Status Register (STAT_REG)	R/W	0x0000_0000	58.3.3.27/58-80
0xBASE+0x006C (TST_MODE_REG)	Test Mode Register (TST_MODE_REG)	R/W	0x0000_0000	58.3.3.28/58-83
0xBASE+0x0070 (USER_MODE_CONT_REG)	User Mode Control Register (USER_MODE_CONT_REG)	R/W	0x0000_0000	58.3.3.29/58-85
0xBASE+0x0074 (SD_TIMING_USR_CONT_REG_0)	SD Timing User Control Register 0 (SD_TIMING_USR_CONT_REG_0)	R/W	0x0000_0000	58.3.3.30/58-86

Table 58-9. TVE Registers (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x0078 (SD_TIMING_USR_CONT_REG_1)	SD Timing User Control Register 1 (SD_TIMING_USR_CONT_REG_1)	R/W	0x0000_0000	58.3.3.31/58-87
0xBASE+0x007C (SD_TIMING_USR_CONT_REG_2)	SD Timing User Control Register 2 (SD_TIMING_USR_CONT_REG_2)	R/W	0x0000_0000	58.3.3.32/58-89
0xBASE+0x0080 (HD_TIMING_USR_CONT_REG_0)	HD Timing User Control Register 0 (HD_TIMING_USR_CONT_REG_0)	R/W	0x0000_0000	58.3.3.33/58-90
0xBASE+0x0084 (HD_TIMING_USR_CONT_REG_1)	HD Timing User Control Register 1 (HD_TIMING_USR_CONT_REG_1)	R/W	0x0000_0000	58.3.3.34/58-91
0xBASE+0x0088 (HD_TIMING_USR_CONT_REG_2)	HD Timing User Control Register 2 (HD_TIMING_USR_CONT_REG_2)	R/W	0x0000_0000	58.3.3.35/58-92
0xBASE+0x008C (LUMA_USR_CONT_REG_0)	Luma User Control Register 0 (LUMA_USR_CONT_REG_0)	R/W	0x0000_0000	58.3.3.36/58-93
0xBASE+0x0090 (LUMA_USR_CONT_REG_1)	Luma User Control Register 1 (LUMA_USR_CONT_REG_1)	R/W	0x0000_0000	58.3.3.37/58-94
0xBASE+0x0094 (LUMA_USR_CONT_REG_2)	Luma User Control Register 2 (LUMA_USR_CONT_REG_2)	R/W	0x0000_0000	58.3.3.38/58-95
0xBASE+0x0098 (LUMA_USR_CONT_REG_3)	Luma User Control Register 3 (LUMA_USR_CONT_REG_3)	R/W	0x0000_0000	58.3.3.39/58-96
0xBASE+0x009C (CSC_USR_CONT_REG_0)	Color Space Conversion User Control Register 0 (CSC_USR_CONT_REG_0)	R/W	0x0000_0000	58.3.3.40/58-97
0xBASE+0x00A0 (CSC_USR_CONT_REG_1)	Color Space Conversion User Control Register 1 (CSC_USR_CONT_REG_1)	R/W	0x0000_0000	58.3.3.41/58-98
0xBASE+0x00A4 (CSC_USR_CONT_REG_2)	Color Space Conversion User Control Register 2 (CSC_USR_CONT_REG_2)	R/W	0x0000_0000	58.3.3.42/58-100
0xBASE+0x00A8 (BLANK_USR_CONT_REG)	Blanking Level User Control Register (BLANK_USR_CONT_REG)	R/W	0x0000_0000	58.3.3.43/58-101
0xBASE+0x00AC (SD_MOD_USR_CONT_REG)	SD Modulation User Control Register (SD_MOD_USR_CONT_REG)	R/W	0x0000_0000	58.3.3.44/58-102

Table 58-9. TVE Registers (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x00B0 (VBI_DATA_USR_CONT_REG_0)	VBI Data User Control Register 0 (VBI_DATA_USR_CONT_REG_0)	R/W	0x0000_0000	58.3.3.45/58-103
0xBASE+0x00B4 (VBI_DATA_USR_CONT_REG_1)	VBI Data User Control Register 1 (VBI_DATA_USR_CONT_REG_1)	R/W	0x0000_0000	58.3.3.46/58-104
0xBASE+0x00B8 (VBI_DATA_USR_CONT_REG_2)	VBI Data User Control Register 2 (VBI_DATA_USR_CONT_REG_2)	R/W	0x0000_0000	58.3.3.47/58-105
0xBASE+0x00BC (VBI_DATA_USR_CONT_REG_3)	VBI Data User Control Register 3 (VBI_DATA_USR_CONT_REG_3)	R/W	0x0000_0000	58.3.3.48/58-106
0xBASE+0x00C0 (VBI_DATA_USR_CONT_REG_4)	VBI Data User Control Register 4 (VBI_DATA_USR_CONT_REG_4)	R/W	0x0000_0000	58.3.3.49/58-107
0xBASE+0x00C4 (DROP_COMP_USR_CONT_REG)	Drop Compensation User Control Register (DROP_COMP_USR_CONT_REG)	R/W	0x0000_0000	58.3.3.50/58-108

58.3.2 Register Summary

The conventions in Figure 58-10 and Table 58-10 serve as a key for the register summary and individual register diagrams.

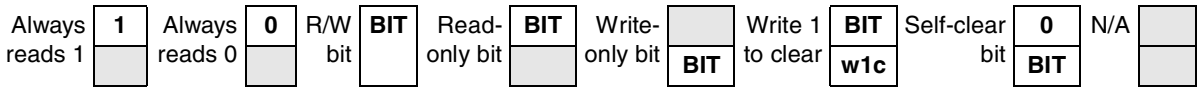


Figure 58-10. Key to Register Fields

Table 58-10 provides a key for register figures and tables and the register summary.

Table 58-10. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.

Table 58-10. Register Conventions (continued)

Convention	Description
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 58-11 is the register summary table.

Table 58-11. Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0000 (COM_CONF_REG)	R	0	0	0	0	0	ACT_LINE_OFFSET[2:0]			0	SYNC_CH_2_EN	SYNC_CH_1_EN	SYNC_CH_0_EN	0	0	SD_PED_AMP_CONT[1:0]	
	W																
	R	0	TV_OUT_MODE[2:0]			TV_STAND[3:0]				P2I_CONV_EN	INP_VIDEO_FORM	DATA_SOURCE_SEL[2:0]		IPU_CLK_EN	TVDAC_SAMP_RATE[1:0]		TVE_EN
	W																
0xBASE+0x0004 (LUMA_FILT_CONT_REG_0)	R	DEFLICK_HIGH_THRESH[7:0]								DEFLICK_MID_THRESH[7:0]							
	W																
	R	DEFLICK_LOW_THRESH[7:0]								0	DEFLICK_COEF[2:0]			0	0	DEFLICK_MEAS_WIN	DEFLICK_EN
	W																
0xBASE+0x0008 (LUMA_FILT_CONT_REG_1)	R	V_SHARP_HIGH_THRESH[7:0]								0	0	0	0	0	0	0	0
	W																
	R	V_SHARP_LOW_THRESH[7:0]								0	V_SHARP_COEF[2:0]			0	0	0	V_SHARP_EN
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000c (LUMA_FILT_CONT_REG_2)	R	H_SHARP_HIGH_THRESH[7:0]								0	0	0	0	0	0	0	0
	W																
	R	H_SHARP_LOW_THRESH[7:0]								0	H_SHARP_COEF[2:0]			0	0	0	H_SHARP_EN
	W																
0xBASE+0x0010 (LUMA_FILT_CONT_REG_3)	R	DERING_HIGH_THRESH[7:0]								DERING_MID_THRESH[7:0]							
	W																
	R	DERING_LOW_THRESH[7:0]								0	DERING_COEF[2:0]			0	SUPP_FILTER_TYPE		DERING_EN
	W																
0xBASE+0x0014 (LUMA_SA_CONT_REG_0)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	SA_V_POINTS_NUM[1:0]		0	0	SA_H_POINTS_NUM[1:0]		0	0	0	LUMA_SA_EN
	W																
0xBASE+0x0018 (LUMA_SA_CONT_REG_1)	R	SA_WIN_V_OFFSET[7:0]								SA_WIN_H_OFFSET[7:0]							
	W																
	R	SA_WIN_HEIGHT[7:0]								SA_WIN_WIDTH[7:0]							
	W																
0xBASE+0x001C (LUMA_SA_STAT_REG_0)	R	DERING_MEAS_MEAN[7:0]								H_SHARP_MEAS_MEAN[7:0]							
	W																
	R	V_SHARP_MEAS_MEAN[7:0]								DEFLICK_MEAS_MEAN[7:0]							
	W																
0xBASE+0x0020 (LUMA_SA_STAT_REG_1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	LUMA_MEAN[7:0]							
	W																
0xBASE+0x0024 (CHROMA_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	SCH_PHASE[7:0]								0	CHROMA_BW[2:0]			0	0	0	CHROMA_V_FILT_EN
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0028 (TVDAC_0_CONT_REG)	R	0	0	0	0	0	0	0	0	BG_RDY_TIME[7:0]							
	W																
	R	TVDAC_0_OFFSET[7:0]								0	0	TVDAC_0_GAIN[5:0]					
	W																
0xBASE+0x002C (TVDAC_1_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	TVDAC_1_OFFSET[7:0]								0	0	TVDAC_1_GAIN[5:0]					
	W																
0xBASE+0x0030 (TVDAC_2_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	TVDAC_2_OFFSET[7:0]								0	0	TVDAC_2_GAIN[5:0]					
	W																
0xBASE+0x0034 (CD_CONT_REG)	R	0	0	0	0	0	0	0	0	0	CD_CH_2_SM_EN	CD_CH_1_SM_EN	CD_CH_0_SM_EN	0	CD_CH_2_LM_EN	CD_CH_1_LM_EN	CD_CH_0_LM_EN
	W																
	R	0	0	0	0	CD_REF_MODE	CD_CH_2_REF_LVL	CD_CH_1_REF_LVL	CD_CH_0_REF_LVL	CD_MON_PER[3:0]				0	0	CD_TRIG_MODE	CD_EN
	W																
0xBASE+0x0038 (VBI_DATA_CONT_REG)	R	0	0	CGMS_HD_B_F2_HEADER[5:0]						0	0	CGMS_HD_B_F1_HEADER[5:0]					
	W																
	R	0	CGMS_HD_B_SW_CRC_EN	CGMS_HD_B_F2_EN	CGMS_HD_B_F1_EN	0	CGMS_HD_A_SW_CRC_EN	CGMS_HD_A_F2_EN	CGMS_HD_A_F1_EN	WSS_SD_EN	CGMS_SD_SW_CRC_EN	CGMS_SD_F2_EN	CGMS_SD_F1_EN	0	CC_SD_BOOST_EN	CC_SD_F2_EN	CC_SD_F1_EN
	W																
0xBASE+0x003C (VBI_DATA_REG_0)	R	0	0	0	0	0	0	0	0	0	0	0	0	CGMS_SD_HD_A_F1_DATA[19:16]			
	W																
	R	CGMS_SD_HD_A_F1_DATA[15:0]															
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0040 (VBI_DATA_REG_1)	R	0	0	0	0	0	0	0	0	0	0	0	0	CGMS_SD_HD_A_F2_DATA[19:16]			
	W																
	R	CGMS_SD_HD_A_F2_DATA[15:0]															
	W																
0xBASE+0x0044 (VBI_DATA_REG_2)	R	CC_SD_CGMS_HD_B_F1_DATA_0[31:16]															
	W																
	R	CC_SD_CGMS_HD_B_F1_DATA_0[15:0]															
	W																
0xBASE+0x0048 (VBI_DATA_REG_3)	R	WSS_SD_CGMS_HD_B_F1_DATA_1[31:16]															
	W																
	R	WSS_SD_CGMS_HD_B_F1_DATA_1[15:0]															
	W																
0xBASE+0x004C (VBI_DATA_REG_4)	R	CGMS_HD_B_F1_DATA_2[31:16]															
	W																
	R	CGMS_HD_B_F1_DATA_2[15:0]															
	W																
0xBASE+0x0050 (VBI_DATA_REG_5)	R	CGMS_HD_B_F1_DATA_3[31:16]															
	W																
	R	CGMS_HD_B_F1_DATA_3[15:0]															
	W																
0xBASE+0x0054 (VBI_DATA_REG_6)	R	CC_SD_CGMS_HD_B_F2_DATA_0[31:16]															
	W																
	R	CC_SD_CGMS_HD_B_F2_DATA_0[15:0]															
	W																
0xBASE+0x0058 (VBI_DATA_REG_7)	R	CGMS_HD_B_F2_DATA_1[31:16]															
	W																
	R	CGMS_HD_B_F2_DATA_1[15:0]															
	W																
0xBASE+0x005C (VBI_DATA_REG_8)	R	CGMS_HD_B_F2_DATA_2[31:16]															
	W																
	R	CGMS_HD_B_F2_DATA_2[15:0]															
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0060 (VBI_DATA_REG_9)	R	CGMS_HD_B_F2_DATA_3[31:16]															
	W																
	R	CGMS_HD_B_F2_DATA_3[15:0]															
	W																
0xBASE+0x0064 (INT_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	SA_MEAS_END_IEN	TVE_FRAME_END_IEN	TVE_FIELD_END_IEN	CGMS_HD_B_F2_DONE_IEN	CGMS_HD_B_F1_DONE_IEN	CGMS_HD_A_F2_DONE_IEN	CGMS_HD_A_F1_DONE_IEN	WSS_SD_DONE_IEN	CGMS_SD_F2_DONE_IEN	CGMS_SD_F1_DONE_IEN	CC_SD_F2_DONE_IEN	CC_SD_F1_DONE_IEN	CD_MON_END_IEN	CD_SM_IEN	CD_LM_IEN
	W																
0xBASE+0x0068 (STAT_REG)	R	0	0	0	0	0	0	BG_READY	CD_MAN_TRIG	0	CD_CH_2_SM_ST	CD_CH_1_SM_ST	CD_CH_0_SM_ST	0	CD_CH_2_LM_ST	CD_CH_1_LM_ST	CD_CH_0_LM_ST
	W								w1s								
	R	0	SA_MEAS_END_INT	TVE_FRAME_END_INT	TVE_FIELD_END_INT	CGMS_HD_B_F2_DONE_INT	CGMS_HD_B_F1_DONE_INT	CGMS_HD_A_F2_DONE_INT	CGMS_HD_A_F1_DONE_INT	WSS_SD_DONE_INT	CGMS_SD_F2_DONE_INT	CGMS_SD_F1_DONE_INT	CC_SD_F2_DONE_INT	CC_SD_F1_DONE_INT	CD_MON_END_INT	CD_SM_INT	CD_LM_INT
	W		w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0xBASE+0x006C (TST_MODE_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	COLORBAR_TYPE
	W																
	R	0	0	TVDAC_TEST_SINE_LEVEL[1:0]		0	TVDAC_TEST_SINE_FREQ[2:0]			0	TVDAC_2_DATA_FORCE	TVDAC_1_DATA_FORCE	TVDAC_0_DATA_FORCE	0	TVDAC_TEST_MODE[2:0]		
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0070 (USER_MODE_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	TVDAC_DROP_COMP_USR_MODE_EN	VBI_DATA_USR_MODE_EN	BLANK_LEVEL_USR_MODE_EN	CSCM_COEF_USR_MODE_EN	SC_FREQ_USR_MODE_EN	LUMA_FILT_USR_MODE_EN	H_TIMING_USR_MODE_EN
	W																
0xBASE+0x0074 (SD_TIMING_USR_CONT_REG_0)	R	0	0	SD_VBI_T2_USR[9:0]										0	0	SD_VBI_T1_USR[9:8]	
	W																
	R	SD_VBI_T1_USR[7:0]								0	0	SD_VBI_T0_USR[5:0]					
	W																
0xBASE+0x0078 (SD_TIMING_USR_CONT_REG_1)	R	0	SD_ACT_T3_USR[6:0]							0	SD_ACT_T2_USR[6:0]						
	W																
	R	0	0	0	SD_ACT_T1_USR[4:0]					0	SD_ACT_T0_USR[6:0]						
	W																
0xBASE+0x007C (SD_TIMING_USR_CONT_REG_2)	R	0	0	SD_ACT_T6_USR[5:0]						0	0	SD_ACT_T5_USR[9:4]					
	W																
	R	0	0	0	SD_ACT_T1_USR[4:0]					0	SD_ACT_T0_USR[6:0]						
	W																
0xBASE+0x0080 (HD_TIMING_USR_CONT_REG_0)	R	0	HD_VBI_T2_USR[10:0]											0	0	0	HD_VBI_T1_USR[8]
	W																
	R	HD_VBI_T1_USR[7:0]								0	HD_VBI_ACT_T0_USR[6:0]						
	W																
0xBASE+0x0084 (HD_TIMING_USR_CONT_REG_1)	R	0	0	0	0	0	0	0	HD_ACT_T1_USR[8:0]								
	W																
	R	0	0	0	HD_VBI_T3_USR[12:0]												
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0088 (HD_TIMING_USR_CONT_REG_2)	R	0	0	0	HD_ACT_T3_USR[12:0]												
	W																
	R	0	0	0	0	HD_ACT_T2_USR[11:0]											
	W																
0xBASE+0x008C (LUMA_USR_CONT_REG_0)	R	0	0	0	0	0	0	0	0	DEFLICK_MASK_MATRIX_USR[23:16]							
	W																
	R	DEFLICK_MASK_MATRIX_USR[15:0]															
	W																
0xBASE+0x0090 (LUMA_USR_CONT_REG_1)	R	0	0	0	0	0	0	0	0	V_SHARP_MASK_MATRIX_USR[23:16]							
	W																
	R	V_SHARP_MASK_MATRIX_USR[15:0]															
	W																
0xBASE+0x0094 (LUMA_USR_CONT_REG_2)	R	0	0	0	0	0	0	0	0	H_SHARP_MASK_MATRIX_USR[23:16]							
	W																
	R	H_SHARP_MASK_MATRIX_USR[15:0]															
	W																
0xBASE+0x0098 (LUMA_USR_CONT_REG_3)	R	0	0	0	0	0	0	0	0	DERING_MASK_MATRIX_USR[23:16]							
	W																
	R	DERING_MASK_MATRIX_USR[15:0]															
	W																
0xBASE+0x009C (CSC_USR_CONT_REG_0)	R	0	0	0	0	0	CSCM_A_COEF_USR[10:0]										
	W																
	R	0	0	BRIGHT_CORR_USR[5:0]						0	0	0	0	0	0	0	DATA_CLIP_USR
	W																
0xBASE+0x00A0 (CSC_USR_CONT_REG_1)	R	0	0	0	0	0	CSCM_C_COEF_USR[10:0]										
	W																
	R	0	0	0	0	CSCM_B_COEF_USR[11:0]											
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x00A4 (CSC_USR_CONT_REG_2)	R	0	0	0	CSCM_E_COEF_USR[12:0]												
	W																
	R	0	0	0	0	CSCM_D_COEF_USR[11:0]											
	W																
0xBASE+0x00A8 (BLANK_USR_CONT_REG)	R	0	0	BLANKING_CH_2_USR[9:0]										BLANKING_CH_1_USR[9:6]			
	W																
	R	BLANKING_CH_1_USR[5:0]						BLANKING_CH_0_USR[9:0]									
	W																
0xBASE+0x00AC (SD_MOD_USR_CONT_REG)	R	0	0	SC_FREQ_USR[24:16]													
	W																
	R	SC_FREQ_USR[15:0]															
	W																
0xBASE+0x00B0 (VBI_DATA_USR_CONT_REG_0)	R	0	0	0	0	VBI_DATA_STOP_TIME_USR[11:0]											
	W																
	R	0	0	0	0	VBI_DATA_START_TIME_USR[11:0]											
	W																
0xBASE+0x00B4 (VBI_DATA_USR_CONT_REG_1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	VBI_PACKET_START_TIME_USR[11:0]											
	W																
0xBASE+0x00B8 (VBI_DATA_USR_CONT_REG_2)	R	0	0	0	0	0	CC_SD_RUNIN_DIV_NUM_USR[10:0]										
	W																
	R	0	0	0	0	CC_SD_RUNIN_START_TIME_USR[11:0]											
	W																
0xBASE+0x00BC (VBI_DATA_USR_CONT_REG_3)	R	0	0	0	CC_SD_CGMS_HD_B_DIV_DENOM_USR[12:0]												
	W																
	R	0	0	0	0	0	0	0	0	0	CC_SD_CGMS_HD_B_DIV_NUM_USR[6:0]						
	W																
0xBASE+0x00C0 (VBI_DATA_USR_CONT_REG_4)	R	0	0	0	WSS_CGMS_SD_CGMS_HD_A_DIV_DENOM_USR[12:0]												
	W																
	R	0	0	0	0	0	0	0	0	0	WSS_CGMS_SD_CGMS_HD_A_DIV_NUM_USR[6:0]						
	W																

Table 58-11. Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x00C4 (DROP_COMP_USR_CONT_REG)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	TVDAC_2_DROP_COMP[3:0]				TVDAC_1_DROP_COMP[3:0]				TVDAC_0_DROP_COMP[3:0]			
	W																

58.3.3 Register Descriptions

This section provides the detailed register descriptions.

58.3.3.1 Common Configuration Register (COM_CONF_REG)

Figure 58-11 shows the common configuration register.

Address 0xBASE+0x0000 (COM_CONF_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	ACT_LINE_OFFSET[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	SYNC_CH_2_EN	SYNC_CH_1_EN	SYNC_CH_0_EN	0	0	SD_PED_AMP_CONT[1:0]	
W								
Reset	0	0	0	1	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	TV_OUT_MODE[2:0]			TV_STAND[3:0]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	P2I_CONV_EN	INP_VIDEO_FORM	DATA_SOURCE_SEL[2:0]		IPU_CLK_EN	TVDAC_SAMP_RATE[1:0]		TVE_EN
W								
Reset	0	0	0	0	0	0	0	0

3

Figure 58-11. Common Configuration Register (COM_CONF_REG)

Register fields are described in Table 58-12.

Table 58-12. Register Field Descriptions

Field	Description
27–31	Reserved.
24–26 ACT_LINE_OFFSET	Offset of the first active video data line relative to a standard value. Defined in lines. 000 No offset 001 1-line offset 010 2-line offset 011 3-line offset 100 3-line offset 101 4-line offset 110 5-line offset 111 7-line offset
23	Reserved.
22 SYNC_CH_2_EN	Enable sync on output channel 2. 0 Disable 1 Enable
21 SYNC_CH_1_EN	Enable sync on output channel 1. 0 Disable 1 Enable
20 SYNC_CH_0_EN	Enable sync on output channel 0. 0 Disable 1 Enable
18–19	Reserved.
16–17 SD_PED_AMP_CONT	Control pedestal, Y/R/G/B black-to-white and sync output amplitude. 00 714 mV (black-to-white)/286 mV (sync), no pedestal 01 714 mV (black-to-white)/286 mV (sync), with pedestal 10 700 mV (black-to-white)/300 mV (sync), no pedestal 11 Reserved
15	Reserved.
12–14 TV_OUT_MODE	Select TV output mode. For HD TV standards, only YPbPr and RGB operation modes are valid. 000 Disable output 001 CVBS on channel #0 010 CVBS on channel #2 011 CVBS on both channels #0 and #2 100 S-video on channels #0 and #1 101 S-video on channels #0 and #1 and CVBS on channel #2 110 YPbPr component 111 RGB component

Table 58-12. Register Field Descriptions (continued)

Field	Description
8–11 TV_STAND	Select TV standard. 0000 SD NTSC 0001 SD PALM 0010 SD Combination PALN 0011 SD PAL (B,D,G,H,I) 0100 HD 720p60 0101 HD 720p50 0110 HD 720p30 0111 HD 720p25 1000 HD 720p24 1001 HD 1080i60 1010 HD 1080i50 1011 HD 1035i60 (1920x1035) 1100 HD 1080p30 1101 HD 1080p25 1110 HD 1080p24 1111 Reserved
7 P2I_CONV_EN	Enable progressive to interlaced conversion. Valid only in for SD TV standards. Must be 0 for HD TV standards. 0 Disable 1 Enable
6 INP_VIDEO_FORM	Select input video format. 0 YCbCr 4:2:2 1 YCbCr 4:4:4
4–5 DATA_SOURCE_SEL	Select data source. 00 Video data bus 1 from the IPU 01 Video data bus 2 from the IPU 10 External test data bus 11 Internal Color Bar Generator
3 IPU_CLK_EN	Enable clock output for the IPU. 0 Disable 1 Enable
1–2 TVDAC_SAMP_RATE	Select TVDAC sampling rate. Must match the TVE_CLK clock frequency. 00 216 MHz (SD mode) or 297 MHz (HD mode) 01 108 MHz (SD mode) or 148.5 MHz (HD mode) 10 54 MHz (SD mode) or reserved (HD mode) 11 Reserved
0 TVE_EN	Enable the TVE. When low, resets all internal registers excluding the IPS registers and gates off internal clocks. 0 Disable 1 Enable

58.3.3.2 Luma Filter Control Register 0 (LUMA_FILT_CONT_REG_0)

Figure 58-12 shows the luma filter control register 0.

Address 0xBASE+0x0004 (LUMA_FILT_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R W	DEFLICK_HIGH_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	DEFLICK_MID_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	DEFLICK_LOW_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	DEFLICK_COEF[2:0]			0	0	DEFLICK_MEAS_WIN	DEFLICK_EN
W								
Reset	0	0	0	0	0	0	0	0

4

Figure 58-12. Luma Filter Control Register 0 (LUMA_FILT_CONT_REG_0)

Register fields are described in Table 58-13.

Table 58-13. Register Field Descriptions

Field	Description
24–31 DEFLICK_LOW_THRESH	Vertical deflickering/ fine sharpening/high-frequency noise reduction high threshold. 000000000 000000011 11111111255
16–23 DEFLICK_MID_THRESH	Vertical deflickering/ fine sharpening/high-frequency noise reduction mid threshold. 000000000 000000011 11111111255
8–15 DEFLICK_LOW_THRESH	Vertical deflickering/ fine sharpening/high-frequency noise reduction low threshold. 000000000 000000011 11111111255

Table 58-13. Register Field Descriptions (continued)

Field	Description
4–6 DEFLICK_COEF	Vertical deflickering/ fine sharpening/high-frequency noise reduction coefficient. 000 0.125 001 0.25 010 0.375 011 0.5 100 0.625 101 0.75 110 0.875 111 1.0
2–3	Reserved.
1 DEFLICK_MEAS_WIN	Select vertical deflickering/ fine sharpening/high-frequency noise reduction measurement window. 0 1-pixel window 1 3-pixels window
0 DEFLICK_EN	Enable vertical deflickering/ fine sharpening/high-frequency noise reduction filter. 0 Disable 1 Enable

58.3.3.3 Luma Filter Control Register 1 (LUMA_FILT_CONT_REG_1)

Figure 58-13 shows the luma filter control register 1.

Address 0xBASE+0x0008 (LUMA_FILT_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R W	V_SHARP_HIGH_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	V_SHARP_LOW_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	V_SHARP_COEF[2:0]			0	0	0	V_SHARP_EN
W								
Reset	0	0	0	0	0	0	0	0

5

Figure 58-13. Luma Filter Control Register 1 (LUMA_FILT_CONT_REG_1)

Register fields are described in Table 58-14.

Table 58-14. Register Field Descriptions

Field	Description
24–31 V_SHARP_LOW_THRESH	Vertical coarse sharpening/ mid-frequency noise reduction high threshold. 000000000 000000011 11111111255
16–23	Reserved.
8–15 V_SHARP_LOW_THRESH	Vertical coarse sharpening/ mid-frequency noise reduction low threshold. 000000000 000000011 11111111255
4–6 V_SHARP_COEF	Vertical coarse sharpening/ mid-frequency noise reduction coefficient. 000 0.125 001 0.25 010 0.375 011 0.5 100 0.625 101 0.75 110 0.875 111 1.0
1–3	Reserved.
0 V_SHARP_EN	Enable vertical coarse sharpening/ mid-frequency noise reduction filter. 0 Disable 1 Enable

58.3.3.4 Luma Filter Control Register 2 (LUMA_FILT_CONT_REG_2)

Figure 58-14 shows the luma filter control register 2.

Address 0xBASE+0x000c (LUMA_FILT_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	H_SHARP_HIGH_THRESH[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	H_SHARP_LOW_THRESH[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	H_SHARP_COEF[2:0]			0	0	0	H_SHARP_EN
W								
Reset	0	0	0	0	0	0	0	0

6

Figure 58-14. Luma Filter Control Register 2 (LUMA_FILT_CONT_REG_2)

Register fields are described in Table 58-15.

Table 58-15. Register Field Descriptions

Field	Description
24–31 H_SHARP_LOW_THRESH	Horizontal coarse sharpening/ mid-frequency noise reduction high threshold. 000000000 000000011 11111111255
16–23	Reserved.
8–15 H_SHARP_LOW_THRESH	Horizontal coarse sharpening/ mid-frequency noise reduction low threshold. 000000000 000000011 11111111255

Table 58-15. Register Field Descriptions (continued)

Field	Description
4–6 H_SHARP_COEF	Horizontal coarse sharpening/ mid-frequency noise reduction coefficient. 000 0.125 001 0.25 010 0.375 011 0.5 100 0.625 101 0.75 110 0.875 111 1.0
1–3	Reserved.
0 H_SHARP_EN	Enable horizontal coarse sharpening/ mid-frequency noise reduction filter. 0 Disable 1 Enable

58.3.3.5 Luma Filter Control Register 3 (LUMA_FILT_CONT_REG_3)

Figure 58-15 shows the luma filter control register 3.

Address 0xBASE+0x0010 (LUMA_FILT_CONT_REG_3) Access: User read/write

	31	30	29	28	27	26	25	24
R W	DERING_HIGH_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	DERING_MID_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	DERING_LOW_THRESH[7:0]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	DERING_COEF[2:0]			0	SUPP_FILTER_TYPE		DERING_EN
W								
Reset	0	0	0	0	0	0	0	0

7

Figure 58-15. Luma Filter Control Register 3 (LUMA_FILT_CONT_REG_3)

Register fields are described in Table 58-16.

Table 58-16. Register Field Descriptions

Field	Description
24–31 DERING_LOW_THRESH	Horizontal deringing/ fine sharpening/high-frequency noise reduction high threshold. 000000000 000000011 11111111255
16–23 DERING_MID_THRESH	Horizontal deringing/ fine sharpening/high-frequency noise reduction mid threshold. 000000000 000000011 11111111255
8–15 DERING_LOW_THRESH	Horizontal deringing/ fine sharpening/high-frequency noise reduction low threshold. 000000000 000000011 11111111255
4–6 DERING_COEF	Horizontal deringing/ fine sharpening/high-frequency noise reduction coefficient. 000 0.125 001 0.25 010 0.375 011 0.5 100 0.625 101 0.75 110 0.875 111 1.0
3	Reserved.
1–2 SUPP_FILTER_TYPE	Select supplement filter type. 00 No supplement filter 01 Lowpass filter 10 PAL notch filter (4.43 MHz) 11 NTSC notch filter (3.58 MHz)
0 DERING_EN	Enable horizontal deringing/ fine sharpening/high-frequency noise reduction filter. 0 Disable 1 Enable

58.3.3.6 Luma Statistic Analysis Control Register 0 (LUMA_SA_CONT_REG_0)

Figure 58-16 shows the luma static analysis control register 0.

Address 0xBASE+0x0014 (LUMA_SA_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	0	0	SA_V_POINTS_NUM[1:0]	
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	SA_H_POINTS_NUM[1:0]		0	0	0	LUMA_SA_EN
W								
Reset	0	0	0	0	0	0	0	0

8

Figure 58-16. Luma Statistic Analysis Control Register 0 (LUMA_SA_CONT_REG_0)

Register fields are described in Table 58-17.

Table 58-17. Register Field Descriptions

Field	Description
10–31	Reserved.
8–9 SA_V_POINTS_NUM	Select number of vertical points (lines) used for statistic analysis. 00 256 points 01 128 points 10 64 points 11 32 points
6–7	Reserved.
4–5 SA_H_POINTS_NUM	Select number of horizontal points (pixels in a line) used for statistic analysis. 00 256 points 01 128 points 10 64 points 11 32 points

Table 58-17. Register Field Descriptions (continued)

Field	Description
1–3	Reserved.
0 LUMA_SA_EN	Enable luma statistic analysis. 0 Disable 1 Enable

58.3.3.7 Luma Statistic Analysis Control Register 1 (LUMA_SA_CONT_REG_1)

Figure 58-17 shows the luma static analysis control register 1.

Address 0xBASE+0x0018 (LUMA_SA_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R W	SA_WIN_V_OFFSET[7:0]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	SA_WIN_H_OFFSET[7:0]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	SA_WIN_HEIGHT[7:0]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	SA_WIN_WIDTH[7:0]							
Reset	0	0	0	0	0	0	0	0

9

Figure 58-17. Luma Statistic Analysis Control Register 1 (LUMA_SA_CONT_REG_1)

Register fields are described in Table 58-18.

Table 58-18. Register Field Descriptions

Field	Description
24–31 SA_WIN_V_OFFSET	Statistic analysis window vertical offset relative to start of an input active video data frame/field. A sum of this parameter and SA_WIN_HEIGHT should not exceed the height of the input video data frame/field. 000000000 lines 000000018 lines ..... 111111112040 lines
16–23 SA_WIN_H_OFFSET	Statistic analysis window horizontal offset relative to start of an input active video data line. A sum of this parameter and SA_WIN_WIDTH should not exceed the length of the input video data line. 000000000 pixels 000000018 pixels ..... 111111112040 pixels
8–15 SA_WIN_HEIGHT	Statistic analysis window height. 000000000 lines 000000018 lines ..... 111111112040 lines
0–7 SA_WIN_WIDTH	Statistic analysis window width. 000000000 pixels 000000018 pixels ..... 111111112040 pixels

58.3.3.8 Luma Statistic Analysis Status Register 0 (LUMA_SA_STAT_REG_0)

Figure 58-18 shows the luma static analysis status register 0.

Address 0xBASE+0x001C (LUMA_SA_STAT_REG_0) Access: User read

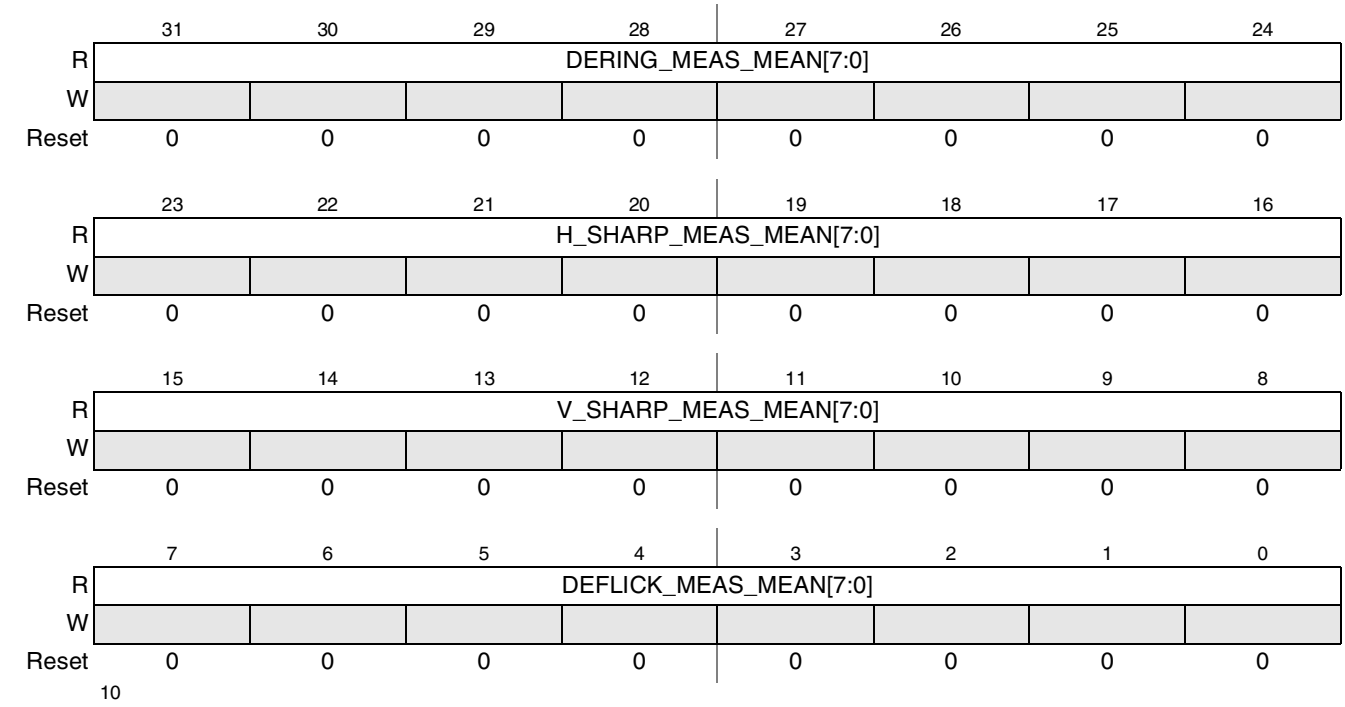


Figure 58-18. Luma Statistic Analysis Status Register 0 (LUMA_SA_STAT_REG_0)

Register fields are described in Table 58-19.

Table 58-19. Register Field Descriptions

Field	Description
24–31 DERING_MEAS_MEAN	Mean absolute value of horizontal deringing/ fine sharpening/high-frequency noise reduction measurement filter output. 000000000 000000011 11111111255
16–23 H_SHARP_MEAS_MEAN	Mean absolute value of horizontal coarse sharpening/ mid-frequency noise reduction measurement filter output. 000000000 000000011 11111111255

Table 58-19. Register Field Descriptions (continued)

Field	Description
8–15 V_SHARP_MEAS_MEAN	Mean absolute value of vertical coarse sharpening/ mid-frequency noise reduction measurement filter output. 000000000 000000011 11111111255
0–7 DEFLICK_MEAS_MEAN	Mean absolute value of vertical deflickering/ fine sharpening/high-frequency noise reduction measurement filter output. 000000000 000000011 11111111255

58.3.3.9 Luma Statistic Analysis Status Register 1 (LUMA_SA_STAT_REG_1)

Figure 58-19 shows the luma static analysis status register 1.

Address 0xBASE+0x0020 (LUMA_SA_STAT_REG_1) Access: User read

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	LUMA_MEAN[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

11

Figure 58-19. Luma Statistic Analysis Status Register 1 (LUMA_SA_STAT_REG_1)

Register fields are described in Table 58-20.

Table 58-20. Register Field Descriptions

Field	Description
8–31	Reserved.
0–7 LUMA_MEAN	Mean value of Luma. 000000000 000000011 11111111255

58.3.3.10 Chroma Control Register (CHROMA_CONT_REG)

Figure 58-20 shows the chroma control register.

Address 0xBASE+0x0024 (CHROMA_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	SCH_PHASE[7:0]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	CHROMA_BW[2:0]			0	0	0	CHROMA_V_FILT_EN
W								
Reset	0	0	1	0	0	0	0	0

12

Figure 58-20. Chroma Control Register (CHROMA_CONT_REG)

Register fields are described in Table 58-21.

Table 58-21. Register Field Descriptions

Field	Description
16–31	Reserved.
8–15 SCH_PHASE	Subcurrier initial phase offset (SD only). The SCH phase is loaded into the 8 MSBs of the phase accumulator and the 22 LSBs are cleared upon reset and every 4 fields for 525-line standards and every 8 fields for 625-line standards. 000000000 degrees 00000001360/256 degrees ..... 11111111255 × 360 ÷ 256 degrees
4–6 CHROMA_BW	Select Chroma filter bandwidth. 000 0.6 MHz for SD mode and YCbCr 4:2:2 input format 1.2 MHz for SD mode and YCbCr 4:4:4 input format 3.3 MHz for HD mode and YCbCr 4:2:2 input format 6.6 MHz for HD mode and YCbCr 4:4:4 input format 001 1.0 MHz for SD mode and YCbCr 4:2:2 input format 2.0 MHz for SD mode and YCbCr 4:4:4 input format 5.5 MHz for HD mode and YCbCr 4:2:2 input format 11.0 MHz for HD mode and YCbCr 4:4:4 input format 010 1.3 MHz for SD mode and YCbCr 4:2:2 input format 3.0 MHz for SD mode and YCbCr 4:4:4 input format 7.1 MHz for HD mode and YCbCr 4:2:2 input format 16.5 MHz for HD mode and YCbCr 4:4:4 input format 011 2.0 MHz for SD mode and YCbCr 4:2:2 input format 4.0 MHz for SD mode and YCbCr 4:4:4 input format 11.0 MHz for HD mode and YCbCr 4:2:2 input format 22.0 MHz for HD mode and YCbCr 4:4:4 input format 100 3.0 MHz for SD mode and YCbCr 4:2:2 input format 6.4 MHz for SD mode and YCbCr 4:4:4 input format 16.5 MHz for HD mode and YCbCr 4:2:2 input format 35.2 MHz for HD mode and YCbCr 4:4:4 input format 101 Reserved 110 Reserved 111 Reserved
1–3	Reserved.
0 CHROMA_V_FILT_EN	Enable vertical chroma filter. Must be 0 for HD mode. 0 Disable 1 Enable

58.3.3.11 TVDAC 0 Control Register (TVDAC_0_CONT_REG)

Figure 58-21 shows the TVDAC 0 control register.

Address 0xBASE+0x0028 (TVDAC_0_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	BG_RDY_TIME[7:0]							
W								
Reset	1	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	TVDAC_0_OFFSET[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	TVDAC_0_GAIN[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

13

Figure 58-21. TVDAC 0 Control Register (TVDAC_0_CONT_REG)

Register fields are described in Table 58-22.

Table 58-22. Register Field Descriptions

Field	Description
24–31	Reserved.
16–23 BG_RDY_TIME	TVDAC bandgap reference ready time. Specified with resolution of 1 msec. This parameter characterizes the TVDAC ready time after enabling the TVDAC. 000000001 msec 000000012 msec ..... 11111110 255 msec 11111111 0 msec

Table 58-22. Register Field Descriptions (continued)

Field	Description
8–15 TVDAC_0_OFFSET	Offset value in the two's complement format for the TVDAC channel #0. Added to the signal after the gain is corrected according to TVDAC_0_GAIN. 10000000-128 10000001-127 ..... 11111111-1 000000000 000000011 01111111127
6–7	Reserved.
0–5 TVDAC_0_GAIN	Select gain value for the TVDAC channel #0. 1000001-32/128 1000011-31/128 ..... 1111111-1/128 0000001 0000011+1/128 ..... 0111111+31/128

58.3.3.12 TVDAC 1 Control Register (TVDAC_1_CONT_REG)

Figure 58-22 shows the TVDAC 1 control register.

Address 0xBASE+0x002C (TVDAC_1_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	1	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	TVDAC_1_OFFSET[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	TVDAC_1_GAIN[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

14

Figure 58-22. TVDAC 1 Control Register (TVDAC_1_CONT_REG)

Register fields are described in Table 58-23.

Table 58-23. Register Field Descriptions

Field	Description
16–31	Reserved.
8–15 TVDAC_1_OFFSET	Offset value in the two's complement format for the TVDAC channel #1. Added to the signal after the gain is corrected according to TVDAC_0_GAIN. 10000000-128 10000001-127 ..... 11111111-1 000000000 000000011 01111111127

Table 58-23. Register Field Descriptions (continued)

Field	Description
6–7	Reserved.
0–5 TVDAC_1_GAIN	Select gain value for the TVDAC channel #1. 1000001-32/128 1000011-31/128 ..... 1111111-1/128 0000001 0000011+1/128 ..... 0111111+31/128

58.3.3.13 TVDAC 2 Control Register (TVDAC_2_CONT_REG)

Figure 58-23 shows the TVDAC 2 control register.

Address 0xBASE+0x0030 (TVDAC_2_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	1	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	TVDAC_2_OFFSET[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	TVDAC_2_GAIN[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

15

Figure 58-23. TVDAC 2 Control Register (TVDAC_2_CONT_REG)

Register fields are described in Table 58-24.

Table 58-24. Register Field Descriptions

Field	Description
16–31	Reserved.
8–15 TVDAC_2_OFFSET	Offset value in the two's complement format for the TVDAC channel #2. Added to the signal after the gain is corrected according to TVDAC_0_GAIN. 10000000-128 10000001-127 ..... 11111111-1 000000000 000000011 01111111127
6–7	Reserved.
0–5 TVDAC_2_GAIN	Select gain value for the TVDAC channel #2. 1000001-32/128 1000011-31/128 ..... 1111111-1/128 0000001 0000011+1/128 ..... 0111111+31/128

58.3.3.14 Cable Detection Control Register (CD_CONT_REG)

Figure 58-24 shows the cable detection control register.

Address 0xBASE+0x0034 (CD_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	CD_CH_2_SM_EN	CD_CH_1_SM_EN	CD_CH_0_SM_EN	0	CD_CH_2_LM_EN	CD_CH_1_LM_EN	CD_CH_0_LM_EN
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	CD_REF_MODE	CD_CH_2_REF_LVL	CD_CH_1_REF_LVL	CD_CH_0_REF_LVL
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	CD_MON_PER[3:0]				0	0	CD_TRIG_MODE	CD_EN
W								
Reset	0	0	0	0	0	0	0	0

16

Figure 58-24. Cable Detection Control Register (CD_CONT_REG)

Register fields are described in Table 58-25.

Table 58-25. Register Field Descriptions

Field	Description
23–31	Reserved.
22 CD_CH_2_SM_EN	TVDAC channel #2 short monitoring enable. 0 Disable 1 Enable
21 CD_CH_1_SM_EN	TVDAC channel #1 short monitoring enable. 0 Disable 1 Enable
20 CD_CH_2_SM_EN	TVDAC channel #0 short monitoring enable. 0 Disable 1 Enable
19	Reserved.
18 CD_CH_2_LM_EN	TVDAC channel #2 load impedance monitoring enable. 0 Disable 1 Enable

Table 58-25. Register Field Descriptions (continued)

Field	Description
17 CD_CH_1_LM_EN	TVDAC channel #1 load impedance monitoring enable. 0 Disable 1 Enable
16 CD_CH_2_LM_EN	TVDAC channel #0 load impedance monitoring enable. 0 Disable 1 Enable
12–15	Reserved.
11 CD_REF_MODE	CD reference mode. Specifies how the reference voltage level for CD comparators is set for active TVDAC channels. The bit is ignored for inactive channels or in standby mode (manual mode assumed). The bit is also ignored for channels# 1 and #2 in YPrPb mode (TV_OUT_MODE=110). 0 Automatic mode when the reference voltage is set according to the channel signal type defined by the TV_OUT_MODE 1 Manual mode when the reference voltage is set according to the CD_CH_#_REF_LVL
10 CD_CH_2_REF_LVL	CD reference level for TVDAC channel #2. Specified the expected level of the channel output voltage when the output is loaded by nominal 37.5-Ohm impedance. This bit is ignored if the channel is active and CD_REF_MODE is 0 excluding YPrPb mode (TV_OUT_MODE=110). 0 The reference level corresponds to the Luma blanking level 1 The reference level corresponds to the Chroma blanking level
8 CD_CH_1_REF_LVL	CD reference level for TVDAC channel #1. Specified the expected level of the channel output voltage when the output is loaded by nominal 37.5-Ohm impedance. This bit is ignored if the channel is active and CD_REF_MODE is 0 excluding YPrPb mode (TV_OUT_MODE=110). 0 The reference level corresponds to the Luma blanking level 1 The reference level corresponds to the Chroma blanking level
8 CD_CH_0_REF_LVL	CD reference level for TVDAC channel #0. Specified the expected level of the channel output voltage when the output is loaded by nominal 37.5-Ohm impedance. This bit is ignored if the channel is active and CD_REF_MODE is 0. 0 The reference level corresponds to the Luma blanking level 1 The reference level corresponds to the Chroma blanking level
4–7 CD_MON_PER	Cable detection monitoring period. If TV_OUT_MODE is not 000 (during normal operation), specified in TV fields for SD or in TV frames for HD. If TV_OUT_MODE is 000 (standby mode), specified in 0.31-sec units. 0000 1 field (SD) or 1 frame (HD) for normal operation or 0.31 sec for standby mode 0001 2 fields (SD) or 2 frames (HD) for normal operation or 0.62 sec for standby mode ..... 1111 16 fields (SD) or 16 frames (HD) for normal operation or 4.96 sec for standby mode
2–3	Reserved.
0 CD_TRIG_MODE	CD trigger mode. 0 Automatic (periodical) trigger 1 Manual (single shot) trigger
0 CD_EN	Enable cable detection. When low, resets the CD Control Unit and the CD Circuit. 0 Disable 1 Enable

58.3.3.15 VBI Data Control Register (VBI_DATA_CONT_REG)

Figure 58-25 shows the VBI data control register.

Address 0xBASE+0x0038 (VBI_DATA_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	CGMS_HD_B_F2_HEADER[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	CGMS_HD_B_F1_HEADER[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	CGMS_HD_B_SW_CRC_EN	CGMS_HD_B_F2_EN	CGMS_HD_B_F1_EN	0	CGMS_HD_A_SW_CRC_EN	CGMS_HD_A_F2_EN	CGMS_HD_A_F1_EN
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	WSS_SD_EN	CGMS_SD_SW_CRC_EN	CGMS_SD_F2_EN	CGMS_SD_F1_EN	0	CC_SD_BOOST_EN	CC_SD_F2_EN	CC_SD_F1_EN
W								
Reset	0	0	0	0	0	0	0	0

17

Figure 58-25. VBI Data Control Register (VBI_DATA_CONT_REG)

Register fields are described in Table 58-26.

Table 58-26. Register Field Descriptions

Field	Description
30–31	Reserved.
24–29 CGMS_HD_B_F2_HEADER	HD CGMS Type B header for field 2.
22–23	Reserved.
16–21 CGMS_HD_B_F1_HEADER	HD CGMS Type B header for field 1.
15	Reserved.
14 CGMS_HD_B_SW_CRC_EN	Enable HD CGMS Type B CRC value calculated by software. Otherwise the CRC value is calculated by hardware. 0 Disable 1 Enable

Table 58-26. Register Field Descriptions (continued)

Field	Description
13 CGMS_HD_B_F2_EN	Enable HD CGMS Type B data insertion for field 2. 0 Disable 1 Enable
12 CGMS_HD_B_F1_EN	Enable HD CGMS Type B data insertion for field 1. 0 Disable 1 Enable
11	Reserved.
10 CGMS_HD_A_SW_CRC_EN	Enable HD CGMS Type A CRC value calculated by software. Otherwise the CRC value is calculated by hardware. 0 Disable 1 Enable
9 CGMS_HD_A_F2_EN	Enable HD CGMS Type A data insertion for field 2. 0 Disable 1 Enable
8 CGMS_HD_A_F1_EN	Enable HD CGMS Type A data insertion for field 1. 0 Disable 1 Enable
7 WSS_SD_EN	Enable SD WSS data insertion. 0 Disable 1 Enable
6 CGMS_SD_SW_CRC_EN	Enable SD CGMS CRC value calculated by software. Otherwise the CRC value is calculated by hardware. 0 Disable 1 Enable
5 CGMS_SD_F2_EN	Enable SD CGMS data insertion for field 2. 0 Disable 1 Enable
4 CGMS_SD_F1_EN	Enable SD CGMS data insertion for field 1. 0 Disable 1 Enable
3	Reserved.
2 CC_SD_BOOST_EN	Enable SD closed caption level boost by a factor of 1.7. May be used in RGB TV output mode. 0 Disable 1 Enable
1 CC_SD_F2_EN	Enable SD closed caption data insertion for field 2. 0 Disable 1 Enable
0 CC_SD_F1_EN	Enable SD closed caption data insertion for field 1. 0 Disable 1 Enable

58.3.3.16 VBI Data Register 0 (VBI_DATA_REG_0)

Figure 58-26 shows the VBI data register 0.

Address 0xBASE+0x003C (VBI_DATA_REG_0) Access: User read/write

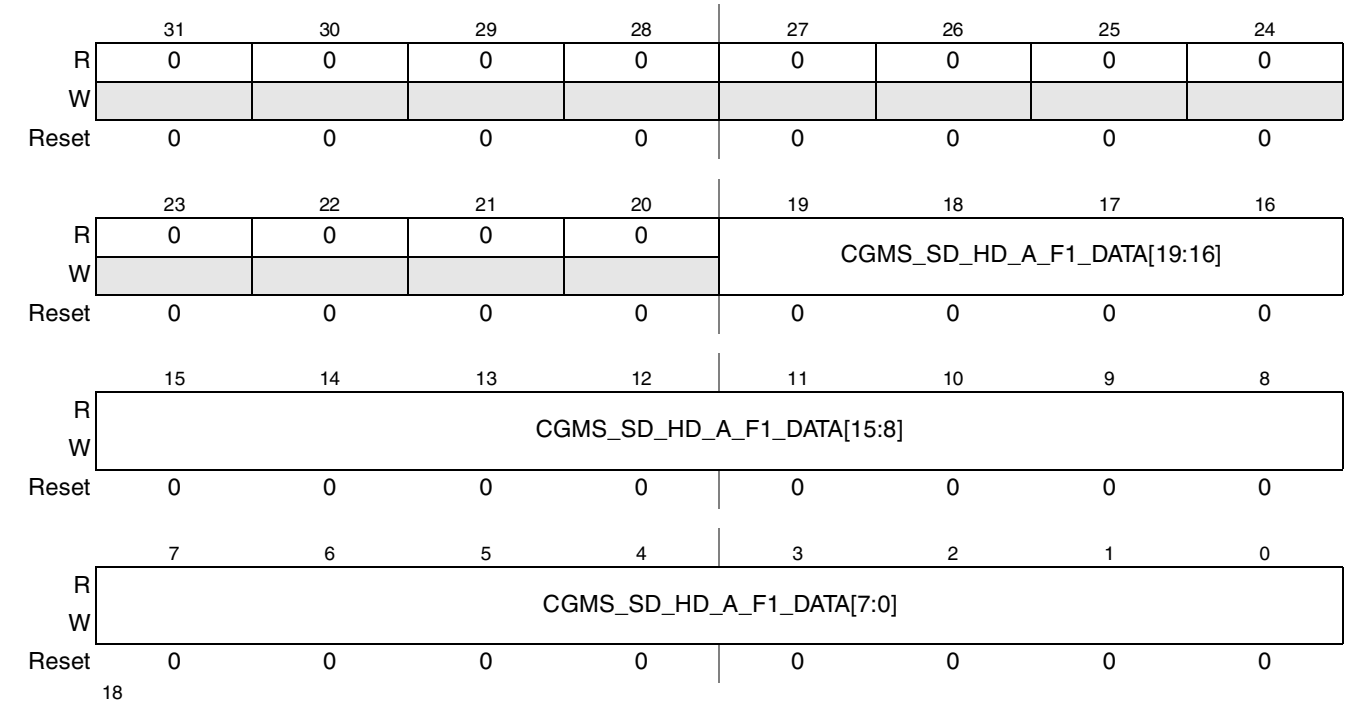


Figure 58-26. VBI Data Register 0 (VBI_DATA_REG_0)

Register fields are described in Table 58-27.

Table 58-27. Register Field Descriptions

Field	Description
20–31	Reserved.
0–19 CGMS_SD_HD_A_F1_DATA	CGMS data (in SD mode) or CGMS Type A data (in HD mode) for field 1.

58.3.3.17 VBI Data Register 1 (VBI_DATA_REG_1)

Figure 58-27 shows the VBI data register 1.

Address 0xBASE+0x0040 (VBI_DATA_REG_1) Access: User read/write

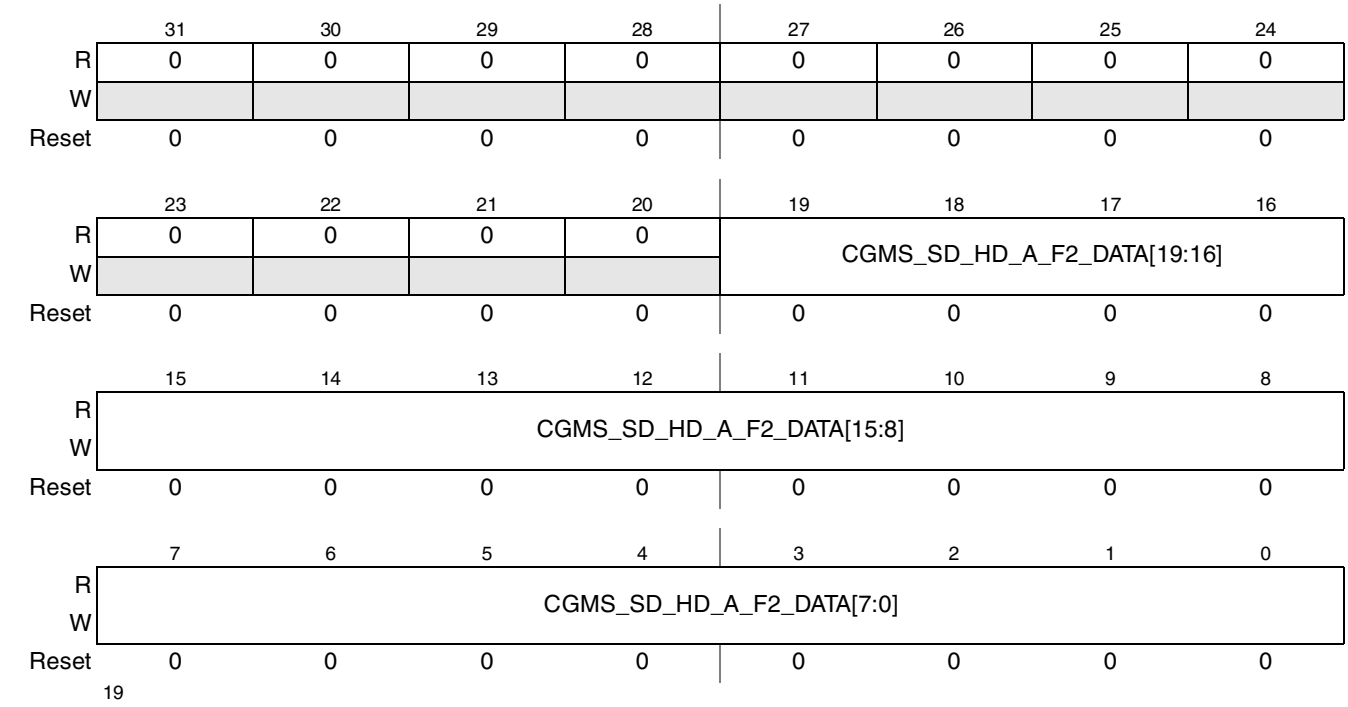


Figure 58-27. VBI Data Register 1 (VBI_DATA_REG_1)

Register fields are described in Table 58-27.

Table 58-28. Register Field Descriptions

Field	Description
20–31	Reserved.
0–19 CGMS_SD_HD_A_F2_DATA	CGMS data (in SD mode) or CGMS Type A data (in HD mode) for field 2.

58.3.3.18 VBI Data Register 2 (VBI_DATA_REG_2)

Figure 58-28 shows the VBI data register 2.

Address 0xBASE+0x0044 (VBI_DATA_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CC_SD_CGMS_HD_B_F1_DATA_0[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CC_SD_CGMS_HD_B_F1_DATA_0[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CC_SD_CGMS_HD_B_F1_DATA_0[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CC_SD_CGMS_HD_B_F1_DATA_0[7:0]							
Reset	0	0	0	0	0	0	0	0

20

Figure 58-28. VBI Data Register 2 (VBI_DATA_REG_2)

Register fields are described in Table 58-29.

Table 58-29. Register Field Descriptions

Field	Description
0–31 CC_SD_CGMS_HD_B_F1_DATA_0	Closed Caption data (in SD mode) or CGMS Type B word 0 data (in HD mode) for field 1. In SD mode only bits CC_SD_CGMS_HD_B_F1_DATA_0[14:8] and CC_SD_CGMS_HD_B_F1_DATA_0[6:0] can be used respectively as MSB and LSB parts of the Closed Caption data.

58.3.3.19 VBI Data Register 3 (VBI_DATA_REG_3)

Figure 58-29 shows the VBI data register 3.

Address 0xBASE+0x0048 (VBI_DATA_REG_3) Access: User read/write

	31	30	29	28	27	26	25	24
R W	WSS_SD_CGMS_HD_B_F1_DATA_1[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	WSS_SD_CGMS_HD_B_F1_DATA_1[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	WSS_SD_CGMS_HD_B_F1_DATA_1[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	WSS_SD_CGMS_HD_B_F1_DATA_1[7:0]							
Reset	0	0	0	0	0	0	0	0

21

Figure 58-29. VBI Data Register 3 (VBI_DATA_REG_3)

Register fields are described in Table 58-30.

Table 58-30. Register Field Descriptions

Field	Description
0–19 WSS_SD_CGMS_HD_B_F1_DATA_1	Wide Screen Signaling data (in SD mode) or CGMS Type B word 1 data (in HD mode) for field 1. In SD mode only bits WSS_SD_CGMS_HD_B_F1_DATA_1[19:0] are used as the WSS data.

58.3.3.20 VBI Data Register 4 (VBI_DATA_REG_4)

Figure 58-30 shows the VBI data register 4.

Address 0xBASE+0x004C (VBI_DATA_REG_4) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CGMS_HD_B_F1_DATA_2[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CGMS_HD_B_F1_DATA_2[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CGMS_HD_B_F1_DATA_2[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CGMS_HD_B_F1_DATA_2[7:0]							
Reset	0	0	0	0	0	0	0	0

22

Figure 58-30. VBI Data Register 4 (VBI_DATA_REG_4)

Register fields are described in Table 58-31.

Table 58-31. Register Field Descriptions

Field	Description
0–19 CGMS_HD_B_F1_DATA_2	CGMS Type B word 2 data (in HD mode) for field 1.

58.3.3.21 VBI Data Register 5 (VBI_DATA_REG_5)

Figure 58-31 shows the VBI data register 5.

Address 0xBASE+0x0050 (VBI_DATA_REG_5) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CGMS_HD_B_F1_DATA_3[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CGMS_HD_B_F1_DATA_3[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CGMS_HD_B_F1_DATA_3[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CGMS_HD_B_F1_DATA_3[7:0]							
Reset	0	0	0	0	0	0	0	0

23

Figure 58-31. VBI Data Register 5 (VBI_DATA_REG_5)

Register fields are described in Table 58-32.

Table 58-32. Register Field Descriptions

Field	Description
0–19 CGMS_HD_B_F1_DATA_3	CGMS Type B word 3 data (in HD mode) for field 1.

58.3.3.22 VBI Data Register 6 (VBI_DATA_REG_6)

Figure 58-32 shows the VBI data register 6.

Address 0xBASE+0x0054 (VBI_DATA_REG_6) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CC_SD_CGMS_HD_B_F2_DATA_0[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CC_SD_CGMS_HD_B_F2_DATA_0[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CC_SD_CGMS_HD_B_F2_DATA_0[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CC_SD_CGMS_HD_B_F2_DATA_0[7:0]							
Reset	0	0	0	0	0	0	0	0

24

Figure 58-32. VBI Data Register 6 (VBI_DATA_REG_6)

Register fields are described in Table 58-33.

Table 58-33. Register Field Descriptions

Field	Description
0–31 CC_SD_CGMS_HD_B_F2_DATA_0	Closed Caption data (in SD mode) or CGMS Type B word 0 data (in HD mode) for field 2. In SD mode only bits CC_SD_CGMS_HD_B_F2_DATA_0[14:8] and CC_SD_CGMS_HD_B_F2_DATA_0[6:0] can be used respectively as MSB and LSB parts of the Closed Caption data.

58.3.3.23 VBI Data Register 7 (VBI_DATA_REG_7)

Figure 58-33 shows the VBI data register 7.

Address 0xBASE+0x0058 (VBI_DATA_REG_7) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CGMS_HD_B_F2_DATA_1[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CGMS_HD_B_F2_DATA_1[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CGMS_HD_B_F2_DATA_1[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CGMS_HD_B_F2_DATA_1[7:0]							
Reset	0	0	0	0	0	0	0	0

25

Figure 58-33. VBI Data Register 7 (VBI_DATA_REG_7)

Register fields are described in Table 58-34.

Table 58-34. Register Field Descriptions

Field	Description
0–19 CGMS_HD_B_F2_DATA_1	CGMS Type B word 1 data (in HD mode) for field 2.

58.3.3.24 VBI Data Register 8 (VBI_DATA_REG_8)

Figure 58-34 shows the VBI data register 8.

Address 0xBASE+0x005C (VBI_DATA_REG_8) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CGMS_HD_B_F2_DATA_2[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CGMS_HD_B_F2_DATA_2[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CGMS_HD_B_F2_DATA_2[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CGMS_HD_B_F2_DATA_2[7:0]							
Reset	0	0	0	0	0	0	0	0

26

Figure 58-34. VBI Data Register 8 (VBI_DATA_REG_8)

Register fields are described in Table 58-35.

Table 58-35. Register Field Descriptions

Field	Description
0–19 CGMS_HD_B_F2_DATA_2	CGMS Type B word 2 data (in HD mode) for field 2.

58.3.3.25 VBI Data Register 9 (VBI_DATA_REG_9)

Figure 58-35 shows the VBI data register 9.

Address 0xBASE+0x0060 (VBI_DATA_REG_9) Access: User read/write

	31	30	29	28	27	26	25	24
R W	CGMS_HD_B_F2_DATA_3[31:24]							
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R W	CGMS_HD_B_F2_DATA_3[23:16]							
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R W	CGMS_HD_B_F2_DATA_3[15:8]							
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R W	CGMS_HD_B_F2_DATA_3[7:0]							
Reset	0	0	0	0	0	0	0	0

27

Figure 58-35. VBI Data Register 5 (VBI_DATA_REG_9)

Register fields are described in Table 58-36.

Table 58-36. Register Field Descriptions

Field	Description
0–19 CGMS_HD_B_F2_DATA_3	CGMS Type B word 3 data (in HD mode) for field 2.

58.3.3.26 Interrupt Control Register (INT_CONT_REG)

Figure 58-36 shows the interrupt control register.

Address 0xBASE+0x0064 (INT_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	SA_MEAS_END_IEN	TVE_FRAME_END_IEN	TVE_FIELD_END_IEN	CGMS_HD_B_F2_DONE_IEN	CGMS_HD_B_F1_DONE_IEN	CGMS_HD_A_F2_DONE_IEN	CGMS_HD_A_F1_DONE_IEN
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	WSS_SD_DONE_IEN	CGMS_SD_F2_DONE_IEN	CGMS_SD_F1_DONE_IEN	CC_SD_F2_DONE_IEN	CC_SD_F1_DONE_IEN	CD_MON_END_IEN	CD_SM_IEN	CD_LM_IEN
W								
Reset	0	0	0	0	0	0	0	0

28

Figure 58-36. Interrupt Control Register (INT_CONT_REG)

Register fields are described in Table 58-37.

Table 58-37. Register Field Descriptions

Field	Description
15–31	Reserved.
14 SA_MEAS_END_IEN	Enable Luma statistic measurement end interrupt. 0 Disable 1 Enable
13 TVE_FRAME_END_IEN	Enable end-of-field interrupt. 0 Disable 1 Enable
12 TVE_FIELD_END_IEN	Enable end-of-frame interrupt. 0 Disable 1 Enable
11 CGMS_HD_B_F2_DONE_IEN	Enable HD CGMS Type B done interrupt for field 2. 0 Disable 1 Enable

Table 58-37. Register Field Descriptions (continued)

Field	Description
10 CGMS_HD_B_F1_DONE_IEN	Enable HD CGMS Type B done interrupt for field 1. 0 Disable 1 Enable
9 CGMS_HD_A_F2_DONE_IEN	Enable HD CGMS Type A done interrupt for field 2. 0 Disable 1 Enable
8 CGMS_HD_A_F1_DONE_IEN	Enable HD CGMS Type A done interrupt for field 1. 0 Disable 1 Enable
7 WSS_SD_DONE_IEN	Enable SD WSS done interrupt. 0 Disable 1 Enable
6 CGMS_SD_F2_DONE_IEN	Enable SD CGMS done interrupt for field 2. 0 Disable 1 Enable
5 CGMS_SD_F1_DONE_IEN	Enable SD CGMS done interrupt for field 1. 0 Disable 1 Enable
4 CC_SD_F2_DONE_IEN	Enable SD closed caption done interrupt for field 2. 0 Disable 1 Enable
3 CC_SD_F1_DONE_IEN	Enable SD closed caption done interrupt for field 1. 0 Disable 1 Enable
2 CD_MON_END_IEN	Enable CD end-of-monitoring interrupt. 0 Disable 1 Enable
1 CD_SM_IEN	Enable CD short monitoring interrupt. 0 Disable 1 Enable
0 CD_LM_IEN	Enable CD load resistance monitoring interrupt. 0 Disable 1 Enable

58.3.3.27 Status Register (STAT_REG)

Figure 58-37 shows the status register.

Address 0xBASE+0x0068 (STAT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	BG_READY	CD_MAN_TRIG
W								w1s
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	CD_CH_2_SM_ST	CD_CH_1_SM_ST	CD_CH_0_SM_ST	0	CD_CH_2_LM_ST	CD_CH_1_LM_ST	CD_CH_0_LM_ST
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	SA_MEAS_END_INT	TVE_FRAME_END_INT	TVE_FIELD_END_INT	CGMS_HD_B_F2_DONE_INT	CGMS_HD_B_F1_DONE_INT	CGMS_HD_A_F2_DONE_INT	CGMS_HD_A_F1_DONE_INT
W		w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	WSS_SD_DONE_INT	CGMS_SD_F2_DONE_INT	CGMS_SD_F1_DONE_INT	CC_SD_F2_DONE_INT	CC_SD_F1_DONE_INT	CD_MON_END_INT	CD_SM_INT	CD_LM_INT
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0

29

Figure 58-37. Status Register (STAT_REG)

Register fields are described in Table 58-25.

Table 58-38. Register Field Descriptions

Field	Description
26–31	Reserved.
25 BG_READY	Status bit of bandgap reference of the TVDAC. The ready time is set by the BG_RDY_TIME parameter. 0 Bandgap reference is not ready. 1 Bandgap reference is ready
24 CD_MAN_TRIG	Control/status bit for manual (single shot) trigger of CD monitors. Set by the MCU, cleared by the TVE. Has no effect if CD_TRIG_MODE is 0. This bit can be set only if CD_EN=1. 0 No CD trigger 1 Trigger CD
23	Reserved.

Table 58-38. Register Field Descriptions (continued)

Field	Description
22 CD_CH_2_SM_ST	Status bit of TVDAC channel #2 short monitoring by CD. Valid only if CD_CH_2_SM_EN is set. 0 Short detected 1 No short detected
21 CD_CH_1_SM_ST	Status bit of TVDAC channel #1 short monitoring by CD. Valid only if CD_CH_1_SM_EN is set. 0 Short detected 1 No short detected
20 CD_CH_0_SM_ST	Status bit of TVDAC channel #0 short monitoring by CD. Valid only if CD_CH_0_SM_EN is set. 0 Short detected 1 No short detected
19	Reserved.
18 CD_CH_2_LM_ST	Status bit of TVDAC channel #2 load impedance monitoring by CD. Valid only if CD_CH_2_LM_EN is set. 0 Cable detected 1 No cable detected
17 CD_CH_1_LM_ST	Status bit of TVDAC channel #1 load impedance monitoring by CD. Valid only if CD_CH_1_LM_EN is set. 0 Cable detected 1 No cable detected
16 CD_CH_0_LM_ST	Status bit of TVDAC channel #0 load impedance monitoring by CD. Valid only if CD_CH_0_LM_EN is set. 0 Cable detected 1 No cable detected
15	Reserved.
14 SA_MEAS_END_INT	Status bit of the Luma statistic measurement end interrupt. 0 No statistic measurement end event has occurred 1 Statistic measurement end event has occurred
13 TVE_FRAME_END_INT	Status bit of the end-of-frame interrupt. 0 No end-of-frame event has occurred 1 End-of-frame event has occurred
12 TVE_FIELD_END_INT	Status bit of the end-of-field interrupt. 0 No end-of-field event has occurred 1 End-of-field event has occurred
11 CGMS_HD_B_F2_DONE_IEN	Status bit of the HD CGMS Type B done interrupt for field 2. 0 No HD CGMS Type B done 1 HD CGMS Type B done
10 CGMS_HD_B_F1_DONE_IEN	Status bit of the HD CGMS Type B done interrupt for field 1. 0 No HD CGMS Type B done 1 HD CGMS Type B done
9 CGMS_HD_A_F2_DONE_IEN	Status bit of the HD CGMS Type A done interrupt for field 2. 0 No HD CGMS Type B done 1 HD CGMS Type B done
8 CGMS_HD_A_F1_DONE_IEN	Status bit of the HD CGMS Type A done interrupt for field 1. 0 No HD CGMS Type B done 1 HD CGMS Type B done
7 WSS_SD_DONE_IEN	Status bit of the SD WSS done interrupt. 0 No SD WSS done 1 SD WSS done

Table 58-38. Register Field Descriptions (continued)

Field	Description
6 CGMS_SD_F2_DONE_IEN	Status bit of the SD CGMS done interrupt for field 2. 0 No SD CGMS done 1 SD CGMS done
5 CGMS_SD_F1_DONE_IEN	Status bit of the SD CGMS done interrupt for field 1. 0 No SD CGMS done 1 SD CGMS done
4 CC_SD_F2_DONE_IEN	Status bit of the SD Closed Caption done interrupt for field 2. 0 No SD Closed Caption done 1 SD Closed Caption done
3 CC_SD_F1_DONE_IEN	Status bit of the SD Closed Caption done interrupt for field 1. 0 No SD Closed Caption done 1 SD Closed Caption done
2 CD_MON_END_INT	Status bit of the end-of-monitoring interrupt by CD. 0 No end-of-monitoring event has occurred 1 End-of-monitoring event has occurred
1 CD_SM_INT	Status bit of the short monitoring interrupt by CD. 0 No short detected 1 Short detected
0 CD_LM_INT	Status bit of the load impedance monitoring interrupt by CD. 0 No mismatch between current output mode and real output load configuration 1 Current output mode does not match real output load configuration

58.3.3.28 Test Mode Register (TST_MODE_REG)

Figure 58-38 shows the test mode register.

Address 0xBASE+0x006C (TST_MODE_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	COLORBAR_TYPE
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	TVDAC_TEST_SINE_LEVEL[1:0]		0	TVDAC_TEST_SINE_FREQ[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	TVDAC_2_DATA_FORCE	TVDAC_1_DATA_FORCE	TVDAC_0_DATA_FORCE	0	TVDAC_TEST_MODE[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

Figure 58-38. Test Mode Register (TST_MODE_REG)

Register fields are described in Table 58-39.

Table 58-39. Register Field Descriptions

Field	Description
17–31	Reserved.
16 COLORBAR_TYPE	Select the type of internally generated color bar. 0 100% 1 75%
14–15	Reserved
8–10 TVDAC_TEST_SINE_LEVEL	Select level of internally generated sine wave pattern. 00 Sine wave swing is 100% of TVDAC full scale (from 1 to 1023) 01 Sine wave swing is 75% of TVDAC full scale 10 Sine wave swing is 50% of TVDAC full scale 11 Sine wave swing is 25% of TVDAC full scale
11	Reserved

Table 58-39. Register Field Descriptions (continued)

Field	Description
8–10 TVDAC_TEST_ SINE_FREQ	Select frequency of internally generated sine wave pattern. Defined ratio (division factor) between the TVDAC sampling frequency and the test sine wave frequency. 000 8 001 16 010 32 011 64 100 128 101 256 110 Reserved 111 Reserved
7	Reserved
6 TVDAC_2_DATA _FORCE	Enable forcing input of TVDAC channel #2 by value defined in BLANKING_CH_2_USR. 0 Disable 1 Enable
5 TVDAC_1_DATA _FORCE	Enable forcing input of TVDAC channel #1 by value defined in BLANKING_CH_1_USR. 0 Disable 1 Enable
4 TVDAC_0_DATA _FORCE	Enable forcing input of TVDAC channel #0 by value defined in BLANKING_CH_0_USR. 0 Disable 1 Enable
3	Reserved
0–2 TVDAC_TEST_ MODE	Select TVDAC test mode. 000 Disable test mode (functional mode) 001 TVDAC test mode 1 - different data on TVDAC channels and UF is in data path 010 TVDAC test mode 2 - equal data on TVDAC channels and UF is in data path 011 TVDAC test mode 3 - different data on TVDAC channels and UF is bypassed 100 TVDAC test mode 4 - equal data on TVDAC channels and UF is bypassed 101 TVDAC test mode 5 - sine wave data from the internal generator directly to all TVDAC channels 110 Reserved 111 Reserved

58.3.3.29 User Mode Control Register (USER_MODE_CONT_REG)

Figure 58-39 shows the user mode control register.

Address 0xBASE+0x0070 (USER_MODE_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	TVDAC_DROP_COMP_USR_MODE_EN	VBI_DATA_USR_MODE_EN	BLANK_LEVEL_USR_MODE_EN	CSCM_COEF_USR_MODE_EN	SC_FREQ_USR_MODE_EN	LUMA_FILT_USR_MODE_EN	H_TIMING_USR_MODE_EN
W								
Reset	0	0	0	0	0	0	0	0

30

Figure 58-39. User Mode Control Register (USER_MODE_CONT_REG)

Register fields are described in Table 58-40.

Table 58-40. Register Field Descriptions

Field	Description
6–31	Reserved.
5 VBI_DATA_USR_MODE_EN	Enable user mode for VBI data. 0 Disable 1 Enable
4 BLANK_LEVEL_USR_MODE_EN	Enable user mode for blank level. 0 Disable 1 Enable
3 CSCM_COEF_USR_MODE_EN	Enable user mode for color space conversion matrix. 0 Disable 1 Enable
2 SC_FREQ_USR_MODE_EN	Enable user mode for subcarrier frequency. 0 Disable 1 Enable

Table 58-40. Register Field Descriptions (continued)

Field	Description
1 LUMA_FILT_USR_MODE_EN	Enable user mode for Luma filtering. 0 Disable 1 Enable
0 H_TIMING_USR_MODE_EN	Enable user mode for horizontal timing. 0 Disable 1 Enable

58.3.3.30 SD Timing User Control Register 0 (SD_TIMING_USR_CONT_REG_0)

Figure 58-40 shows the SD timing user control register 0.

Address 0xBASE+0x0074 (SD_TIMING_USR_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	SD_VBI_T2_USR[9:4]					
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	SD_VBI_T2_USR[3:0]				0	0	SD_VBI_T1_USR[9:8]	
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	SD_VBI_T1_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	SD_VBI_T0_USR[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

31

Figure 58-40. SD Timing User Control Register 0 (SD_TIMING_USR_CONT_REG_0)

Register fields are described in Table 58-41.

Table 58-41. Register Field Descriptions

Field	Description
30–31	Reserved.
20–29 SD_VBI_T1_USR	User defined horizontal timing interval t2 for VBI lines in SD mode. 00000000001 pixels 00000000012 pixels ..... 11111111111024 pixels

Table 58-41. Register Field Descriptions (continued)

Field	Description
18–19	Reserved.
8–17 SD_VBI_T1_USR	User defined horizontal timing interval t1 for VBI lines in SD mode. 00000000001 pixels 00000000012 pixels ..... 11111111111024 pixels
6–7	Reserved.
0–5 SD_VBI_T0_USR	User defined horizontal timing interval t0 for VBI lines in SD mode. 0000001 pixels 0000012 pixels ..... 11111164 pixels

58.3.3.31 SD Timing User Control Register 1 (SD_TIMING_USR_CONT_REG_1)

Figure 58-41 shows the SD timing user control register 1.

Address 0xBASE+0x0078 (SD_TIMING_USR_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	SD_ACT_T3_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	SD_ACT_T2_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	SD_ACT_T1_USR[4:0]				
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	SD_ACT_T0_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

32

Figure 58-41. SD Timing User Control Register 1 (SD_TIMING_USR_CONT_REG_1)

Register fields are described in Table 58-42.

Table 58-42. Register Field Descriptions

Field	Description
31	Reserved.
24–30 SD_ACT_T3_USR	User defined horizontal timing interval t3 for active data lines in SD mode. 00000001 pixels 00000012 pixels ..... 1111111128 pixels
23	Reserved.
16–22 SD_ACT_T2_USR	User defined horizontal timing interval t2 for active data lines in SD mode. 00000001 pixels 00000012 pixels ..... 1111111128 pixels
13–15	Reserved.
8–12 SD_ACT_T1_USR	User defined horizontal timing interval t1 for active data lines in SD mode. 000001 pixels 000012 pixels ..... 1111132 pixels
7	Reserved.
0–6 SD_ACT_T0_USR	User defined horizontal timing interval t0 for active data lines in SD mode. 00000001 pixels 00000012 pixels ..... 1111111128 pixels

58.3.3.32 SD Timing User Control Register 2 (SD_TIMING_USR_CONT_REG_2)

Figure 58-42 shows the SD timing user control register 2.

Address 0xBASE+0x007C (SD_TIMING_USR_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	SD_ACT_T6_USR[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	SD_ACT_T5_USR[9:4]					
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	SD_ACT_T5_USR[3:0]				0	SD_ACT_T4_USR[10:8]		
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	SD_ACT_T4_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

33

Figure 58-42. SD Timing User Control Register 2 (SD_TIMING_USR_CONT_REG_2)

Register fields are described in Table 58-43.

Table 58-43. Register Field Descriptions

Field	Description
30–31	Reserved.
24–29 SD_ACT_T6_USR	User defined horizontal timing interval t6 for active data lines in SD mode. 0000001 pixels 0000012 pixels ..... 11111164 pixels
22–23	Reserved.
12–21 SD_ACT_T5_USR	User defined horizontal timing interval t5 for active data lines in SD mode. 00000000001 pixels 00000000012 pixels ..... 11111111111024 pixels

Table 58-43. Register Field Descriptions (continued)

Field	Description
11	Reserved.
0–10 SD_ACT_T4_USR	User defined horizontal timing interval t4 for active data lines in SD mode. 000000000001 pixels 000000000012 pixels ..... 111111111112048 pixels

58.3.3.33 HD Timing User Control Register 0 (HD_TIMING_USR_CONT_REG_0)

Figure 58-43 shows the HD timing user control register 0.

Address 0xBASE+0x0080 (HD_TIMING_USR_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	HD_VBI_T2_USR[10:4]						
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	HD_VBI_T2_USR[3:0]				0	0	0	HD_VBI_T1_USR[8]
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	HD_VBI_T1_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	HD_VBI_ACT_T0_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

34

Figure 58-43. HD Timing User Control Register 0 (HD_TIMING_USR_CONT_REG_0)

Register fields are described in Table 58-44.

Table 58-44. Register Field Descriptions

Field	Description
31	Reserved.
20–30 HD_VBI_T2_USR	User defined horizontal timing interval t2 for VBI lines in HD mode. 000000000001 pixels 000000000012 pixels ..... 111111111112048 pixels

Table 58-44. Register Field Descriptions (continued)

Field	Description
17–19	Reserved.
8–16 HD_VBI_T1_USR	User defined horizontal timing interval t1 for VBI lines in HD mode. 0000000001 pixels 0000000012 pixels ..... 111111111512 pixels
7	Reserved.
0–6 HD_VBI_ACT_T0_USR	User defined horizontal timing interval t0 for both VBI and active data lines in HD mode. 00000001 pixels 00000012 pixels ..... 1111111128 pixels

58.3.3.34 HD Timing User Control Register 1 (HD_TIMING_USR_CONT_REG_1)

Figure 58-44 shows the HD timing user control register 1.

Address 0xBASE+0x0084 (HD_TIMING_USR_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	HD_ACT_T1_USR[8]
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	HD_ACT_T1_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	HD_VBI_T3_USR[12:8]				
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	HD_VBI_T3_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

35

Figure 58-44. HD Timing User Control Register 1 (HD_TIMING_USR_CONT_REG_1)

Register fields are described in Table 58-45.

Table 58-45. Register Field Descriptions

Field	Description
25–31	Reserved.
16–24 HD_ACT_T1_USR	User defined horizontal timing interval t1 for active data lines in HD mode. 0000000001 pixels 0000000012 pixels ..... 111111111512 pixels
13–15	Reserved.
0–12 HD_VBI_T3_USR	User defined horizontal timing interval t3 for VBI lines in HD mode. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels

58.3.3.35 HD Timing User Control Register 2 (HD_TIMING_USR_CONT_REG_2)

Figure 58-45 shows the HD timing user control register 2.

Address 0xBASE+0x0088 (HD_TIMING_USR_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	HD_ACT_T3_USR[12:8]				
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	HD_ACT_T3_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	HD_ACT_T2_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	HD_ACT_T2_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

36

Figure 58-45. HD Timing User Control Register 2 (HD_TIMING_USR_CONT_REG_2)

Register fields are described in Table 58-46.

Table 58-46. Register Field Descriptions

Field	Description
29–31	Reserved.
16–28 HD_ACT_T3_USR	User defined horizontal timing interval t1 for active data lines in HD mode. 00000000000001 pixels 00000000000012 pixels ..... 1111111111118192 pixels
12–15	Reserved.
0–11 HD_ACT_T2_USR	User defined horizontal timing interval t3 for active data lines in HD mode. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels

58.3.3.36 Luma User Control Register 0 (LUMA_USR_CONT_REG_0)

Figure 58-46 shows the Luma user control register 0.

Address 0xBASE+0x008C (LUMA_USR_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	DEFLICK_MASK_MATRIX_USR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	DEFLICK_MASK_MATRIX_USR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	DEFLICK_MASK_MATRIX_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

37

Figure 58-46. Luma User Control Register 0 (LUMA_USR_CONT_REG_0)

Register fields are described in Table 58-47.

Table 58-47. Register Field Descriptions

Field	Description
24–31	Reserved.
0–23 DEFLICK_MASK_MATRIX_USR	User defined decision matrix for vertical deflickering/ fine sharpening/high-frequency noise reduction luma filter.

58.3.3.37 Luma User Control Register 1 (LUMA_USR_CONT_REG_1)

Figure 58-47 shows the Luma user control register 1.

Address 0xBASE+0x0090 (LUMA_USR_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	V_SHARP_MASK_MATRIX_USR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	V_SHARP_MASK_MATRIX_USR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	V_SHARP_MASK_MATRIX_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

38

Figure 58-47. Luma User Control Register 1 (LUMA_USR_CONT_REG_1)

Register fields are described in Table 58-48.

Table 58-48. Register Field Descriptions

Field	Description
24–31	Reserved.
0–23 V_SHARP_MASK_MATRIX_USR	User defined decision matrix for vertical coarse sharpening/mid-frequency noise reduction luma filter.

58.3.3.38 Luma User Control Register 2 (LUMA_USR_CONT_REG_2)

Figure 58-48 shows the Luma user control register 2.

Address 0xBASE+0x0094 (LUMA_USR_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	H_SHARP_MASK_MATRIX_USR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	H_SHARP_MASK_MATRIX_USR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	H_SHARP_MASK_MATRIX_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

39

Figure 58-48. Luma User Control Register 2 (LUMA_USR_CONT_REG_2)

Register fields are described in Table 58-49.

Table 58-49. Register Field Descriptions

Field	Description
24–31	Reserved.
0–23 H_SHARP_MASK_MATRIX_USR	User defined decision matrix for horizontal coarse sharpening/mid-frequency noise reduction luma filter.

58.3.3.39 Luma User Control Register 3 (LUMA_USR_CONT_REG_3)

Figure 58-49 shows the Luma user control register 3.

Address 0xBASE+0x0098 (LUMA_USR_CONT_REG_3) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	DERING_MASK_MATRIX_USR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	DERING_MASK_MATRIX_USR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	DERING_MASK_MATRIX_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

40

Figure 58-49. Luma User Control Register 2 (LUMA_USR_CONT_REG_2)

Register fields are described in Table 58-50.

Table 58-50. Register Field Descriptions

Field	Description
24–31	Reserved.
0–23 DERING_MASK_MATRIX_USR	User defined decision matrix for horizontal dringing/ fine sharpening/high-frequency noise reduction luma filter.

58.3.3.40 Color Space Conversion User Control Register 0 (CSC_USR_CONT_REG_0)

Figure 58-50 shows the color space conversion user control register 0.

Address 0xBASE+0x009C (CSC_USR_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	CSCM_A_COEF_USR[10:8]		
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	CSCM_A_COEF_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	BRIGHT_CORR_USR[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	DATA_CLIP_USR
W								
Reset	0	0	0	0	0	0	0	0

41

Figure 58-50. Color Space Conversion User Control Register 0 (CSC_USR_CONT_REG_0)

Register fields are described in Table 58-51.

Table 58-51. Register Field Descriptions

Field	Description
27–31	Reserved.
16–26 CSCM_A_COEF_USR	User defined CSC coefficient A. Represented in unsigned format with 10 fractional bits and 1 integer bit. 000000000000 000000000011/1024 ..... 111111111111+1023/1024
14–15	Reserved
8–13 BRIGHT_CORR_USR	User defined brightness correction offset in two's complement format. 100000-32 100001-31 ..... 0000000 01111131

Table 58-51. Register Field Descriptions (continued)

Field	Description
1–7	Reserved
0 DATA_CLIP_US R	User defined lower clipping level for CSC output data. Relevant only for Luma, composite or RGB ouputs. 0 –192 1 –128

58.3.3.41 Color Space Conversion User Control Register 1 (CSC_USR_CONT_REG_1)

Figure 58-51 shows the color space conversion user control register 1.

Address 0xBASE+0x00A0 (CSC_USR_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	CSCM_C_COEF_USR[10:8]		
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	CSCM_C_COEF_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	CSCM_B_COEF_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	CSCM_B_COEF_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

42

Figure 58-51. Color Space Conversion User Control Register 1 (CSC_USR_CONT_REG_1)

Register fields are described in Table 58-52.

Table 58-52. Register Field Descriptions

Field	Description
27–31	Reserved.
16–26 CSCM_C_COEF_USR	User defined CSC coefficient C. Represented in signed two's complement format with 10 fractional bits and 1 sign bit. 10000000000-1 10000000001-1+1/1024 ..... 11111111111-1/1024 000000000000 000000000011/1024 ..... 011111111111023/1024
12–15	Reserved.
0–11 CSCM_B_COEF_USR	User defined CSC coefficient B. Represented in signed two's complement format with 10 fractional bits, 1 integer bit and 1 sign bit. 100000000000-2 100000000001-2+1/1024 ..... 111111111111-1/1024 0000000000000 0000000000011/1024 ..... 0111111111111+1023/1024

58.3.3.42 Color Space Conversion User Control Register 2 (CSC_USR_CONT_REG_2)

Figure 58-52 shows the color space conversion user control register 2.

Address 0xBASE+0x00A4 (CSC_USR_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	CSCM_E_COEF_USR[12:8]				
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	CSCM_E_COEF_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	CSCM_D_COEF_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	CSCM_D_COEF_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

43

Figure 58-52. Color Space Conversion User Control Register 2 (CSC_USR_CONT_REG_2)

Register fields are described in Table 58-53.

Table 58-53. Register Field Descriptions

Field	Description
29–31	Reserved.
16–28 CSCM_E_COEF_USR	User defined CSC coefficient E. Represented in unsigned format with 10 fractional bits and 3 integer bits. 00000000000000 00000000000011/1024 ..... 11111111111117+1023/1024
12–15	Reserved.
0–11 CSCM_D_COEF_USR	User defined CSC coefficient D. Represented in unsigned format with 10 fractional bits and 2 integer bits. 0000000000000 0000000000011/1024 ..... 1111111111113+1023/1024

58.3.3.43 Blanking Level User Control Register (BLANK_USR_CONT_REG)

Figure 58-53 shows the blanking level user control register.

Address 0xBASE+0x00A8 (BLANK_USR_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	BLANKING_CH_2_USR[9:4]					
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	BLANKING_CH_2_USR[3:0]				BLANKING_CH_1_USR[9:6]			
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	BLANKING_CH_1_USR[5:0]						BLANKING_CH_0_USR[9:8]	
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	BLANKING_CH_0_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

44

Figure 58-53. Blanking Level User Control Register (BLANK_USR_CONT_REG)

Register fields are described in Table 58-54.

Table 58-54. Register Field Descriptions

Field	Description
30–31	Reserved.
20–29 BLANKING_CH_2_USR	User defined blanking level for channel #2. 00000000000 00000000011 11111111111023
10–19 BLANKING_CH_1_USR	User defined blanking level for channel #1. 00000000000 00000000011 11111111111023
0–9 BLANKING_CH_0_USR	User defined blanking level for channel #0. 00000000000 00000000011 11111111111023

58.3.3.44 SD Modulation User Control Register (SD_MOD_USR_CONT_REG)

Figure 58-54 shows the SD modulation user control register.

Address 0xBASE+0x00AC (SD_MOD_USR_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	SC_FREQ_USR[24:29]					
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	SC_FREQ_USR[23:16]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	SC_FREQ_USR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	SC_FREQ_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

45

Figure 58-54. SD Modulation User Control Register (SD_MOD_USR_CONT_REG)

Register fields are described in Table 58-55.

Table 58-55. Register Field Descriptions

Field	Description
30–31	Reserved.
0–29 SC_FREQ_USR	User defined subcarrier frequency factor. The real frequency is SC_FREQ_USR $\times$ 27/2^{30} MHz.

58.3.3.45 VBI Data User Control Register 0 (VBI_DATA_USR_CONT_REG_0)

Figure 58-55 shows the VBI data user control register 0.

Address 0xBASE+0x00B0 (VBI_DATA_USR_CONT_REG_0) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	VBI_DATA_STOP_TIME_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	VBI_DATA_STOP_TIME_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	VBI_DATA_START_TIME_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	VBI_DATA_START_TIME_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

46

Figure 58-55. VBI Data User Control Register 0 (VBI_DATA_USR_CONT_REG_0)

Register fields are described in Table 58-56.

Table 58-56. Register Field Descriptions

Field	Description
30–31	Reserved.
0–11 VBI_DATA_STOP_TIME_USR	User defined stop time for switching from VBI data mode to video mode. Defined relative to line start point. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels
12–15	Reserved.
0–11 VBI_DATA_START_TIME_USR	User defined start time for switching from video mode to VBI data mode. Defined relative to line start point. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels

58.3.3.46 VBI Data User Control Register 1 (VBI_DATA_USR_CONT_REG_1)

Figure 58-56 shows the VBI data user control register 1.

Address 0xBASE+0x00B4 (VBI_DATA_USR_CONT_REG_1) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	VBI_PACKET_START_TIME_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	VBI_PACKET_START_TIME_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

47

Figure 58-56. VBI Data User Control Register 1 (VBI_DATA_USR_CONT_REG_1)

Register fields are described in Table 58-57.

Table 58-57. Register Field Descriptions

Field	Description
30–31	Reserved.
0–11 VBI_PACKET_START_TIME_USR	User defined VBI data packet start time. Defined relative to line start point. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels

58.3.3.47 VBI Data User Control Register 2 (VBI_DATA_USR_CONT_REG_2)

Figure 58-57 shows the VBI data user control register 2.

Address 0xBASE+0x00B8 (VBI_DATA_USR_CONT_REG_2) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	CC_SD_RUNIN_DIV_NUM_USR[10:8]		
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	CC_SD_RUNIN_DIV_NUM_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	CC_SD_RUNIN_START_TIME_USR[11:8]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	CC_SD_RUNIN_START_TIME_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

48

Figure 58-57. VBI Data User Control Register 2 (VBI_DATA_USR_CONT_REG_2)

Register fields are described in Table 58-58.

Table 58-58. Register Field Descriptions

Field	Description
27–31	Reserved.
16–26 CC_SD_RUNIN_DIV_NUM_USR	User defined numerator of the run-in frequency division factor for Closed Caption in SD mode. 000000000000 000000000011 111111111112047
12–15	Reserved.
0–11 CC_SD_RUNIN_START_TIME_USR	User defined run-in sequence start time for Closed Caption in SD mode. Defined relative to line start point. 0000000000001 pixels 0000000000012 pixels ..... 1111111111114096 pixels

58.3.3.48 VBI Data User Control Register 3 (VBI_DATA_USR_CONT_REG_3)

Figure 58-58 shows the VBI data user control register 3.

Address 0xBASE+0x00BC (VBI_DATA_USR_CONT_REG_3) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	CC_SD_CGMS_HD_B_DIV_DENOM_USR[12:8]				
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	CC_SD_CGMS_HD_B_DIV_DENOM_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	CC_SD_CGMS_HD_B_DIV_NUM_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

49

Figure 58-58. VBI Data User Control Register 2 (VBI_DATA_USR_CONT_REG_2)

Register fields are described in Table 58-59.

Table 58-59. Register Field Descriptions

Field	Description
29–31	Reserved.
16–28 CC_SD_CGMS_HD_B_DIV_DENOM_USR	User defined denominator of the data rate division factor for Closed Caption in SD mode and CGMS Type B in HD mode. 00000000000000 00000000000011 11111111111118191
7–15	Reserved.
0–6 CC_SD_CGMS_HD_B_DIV_NUM_USR	User defined numerator of the data rate division factor for Closed Caption in SD mode and CGMS Type B in HD mode. 00000000 00000011 ..... 1111111 127

58.3.3.49 VBI Data User Control Register 4 (VBI_DATA_USR_CONT_REG_4)

Figure 58-59 shows the VBI data user control register 4.

Address 0xBASE+0x00C0 (VBI_DATA_USR_CONT_REG_4) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	WSS_CGMS_SD_CGMS_HD_A_DIV_DENOM_USR[12:8]				
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	WSS_CGMS_SD_CGMS_HD_A_DIV_DENOM_USR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	0	WSS_CGMS_SD_CGMS_HD_A_DIV_NUM_USR[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

50

Figure 58-59. VBI Data User Control Register 2 (VBI_DATA_USR_CONT_REG_2)

Register fields are described in Table 58-60.

Table 58-60. Register Field Descriptions

Field	Description
29–31	Reserved.
16–28 CC_SD_CGMS_HD_B_DIV_DENOM_USR	User defined denominator of the data rate division factor for WSS/CGMS in SD mode and CGMS Type A in HD mode. 00000000000000 00000000000011 11111111111118191
7–15	Reserved.
0–6 WSS_CGMS_SD_CGMS_HD_A_DIV_NUM_USR	User defined numerator of the data rate division factor for WSS/CGMS in SD mode and CGMS Type A in HD mode. 00000000 00000011 ..... 1111111 127

58.3.3.50 Drop Compensation User Control Register (DROP_COMP_USR_CONT_REG)

Figure 58-60 shows the drop compensation user control register.

Address 0xBASE+0x00C4 (DROP_COMP_USR_CONT_REG) Access: User read/write

	31	30	29	28	27	26	25	24
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8
R	0	0	0	0	TVDAC_2_DROP_COMP[3:0]			
W								
Reset	0	0	0	0	0	0	0	0

	7	6	5	4	3	2	1	0
R	TVDAC_1_DROP_COMP[3:0]				TVDAC_0_DROP_COMP[3:0]			
W								
Reset	0	0	0	0	0	0	0	0

51

Figure 58-60. Drop Compensation User Control Register (DROP_COMP_USR_CONT_REG)

Register fields are described in Table 58-60.

Table 58-61. Register Field Descriptions

Field	Description
9–31	Reserved.
8–11 TVDAC_1_DROP_COMP	User defined frequency response drop compensation coefficient K_{dc} for the TVDAC channel #2. 0000 0 0001 1/256 ..... 1111 15/256

Table 58-61. Register Field Descriptions (continued)

Field	Description
4–7 TVDAC_1_DROP_COMP	User defined frequency response drop compensation coefficient K_{dc} for the TVDAC channel #1. 0000 0 0001 1/256 ..... 1111 15/256
0–3 TVDAC_0_DROP_COMP	User defined frequency response drop compensation coefficient K_{dc} for the TVDAC channel #0. 0000 0 0001 1/256 ..... 1111 15/256

Chapter 59
Universal Asynchronous Receiver/Transmitter (UART)

59.1 Overview

This section briefly introduces the universal asynchronous receiver/transmitter (UART) module by showing a top-level diagram of the functional organization of the module, including all off-chip signals. The full description of the module is in Section 59.4, Functional Description."

Figure 59-1 shows the block diagram.

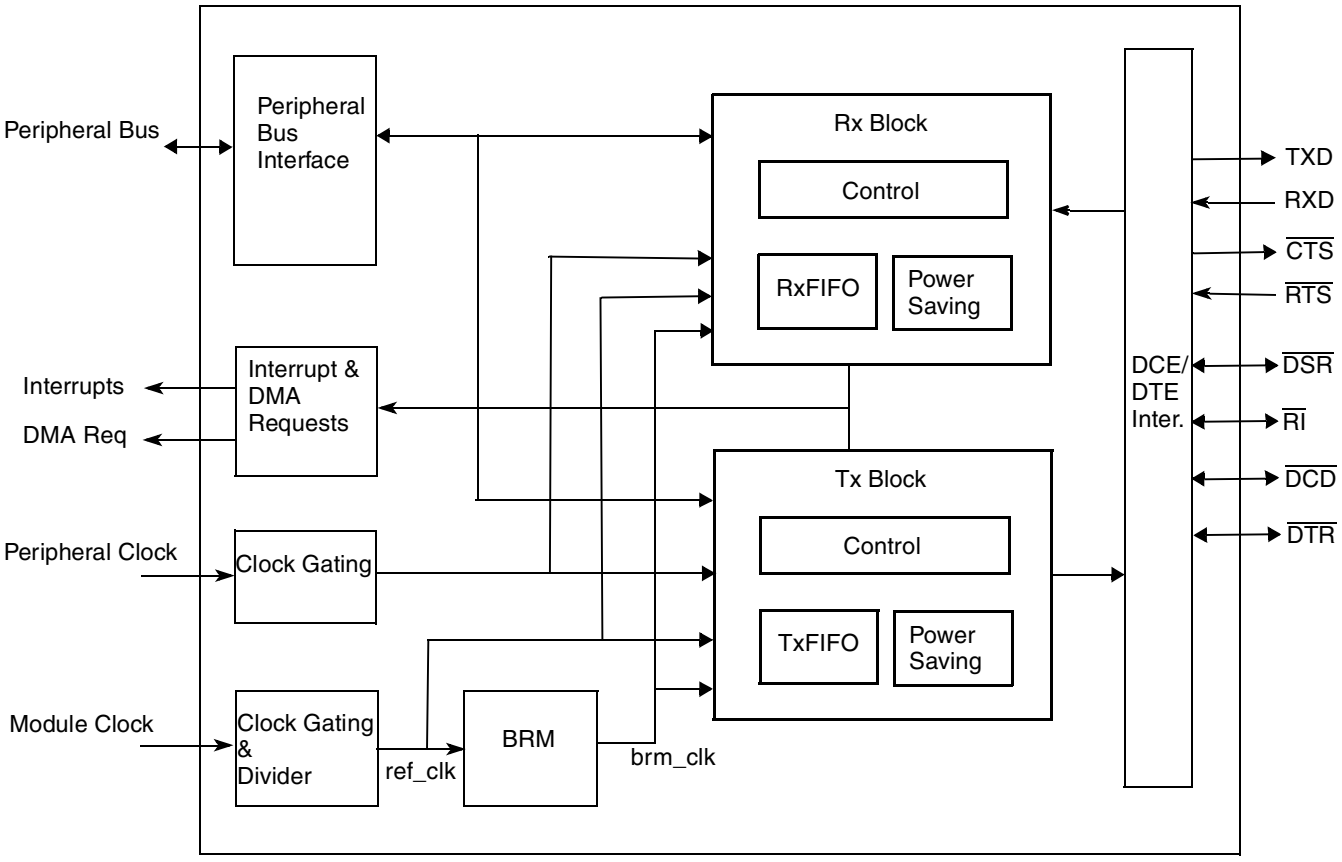


Figure 59-1. UART Block Diagram

59.1.1 Features

The UART includes the following features:

- High-speed TIA/EIA-232-F compatible
- 7 or 8 data bits
- 1 or 2 stop bits
- Programmable parity (even, odd, and no parity)
- Hardware flow control support for request to send ($\overline{\text{RTS}}$) and clear to send ($\overline{\text{CTS}}$) signals
- Edge-selectable $\overline{\text{RTS}}$ and edge-detect interrupts
- Status flags for various flow control and FIFO states
- Serial IR interface low-speed, IrDA-compatible (up to 115.2 Kbit/s).
- Voting logic for improved noise immunity (16× oversampling)
- Transmitter FIFO empty interrupt suppression
- UART internal clocks enable/disable
- Auto baud rate detection (up to 115.2 Kbit/s)
- Receiver and transmitter enable/disable for power saving
- DCE/DTE capability
- $\overline{\text{RTS}}$, IrDA asynchronous wake (AIRINT), receive asynchronous wake (AWAKE), $\overline{\text{RI}}$ (DTE only), $\overline{\text{DCD}}$ (DTE only), $\overline{\text{DTR}}$ (DCE only) and $\overline{\text{DSR}}$ (DTE only) interrupts wake the processor from STOP mode
- Maskable interrupts
- Two DMA Requests (TxFIFO DMA Request and RxFIFO DMA Request)
- Escape character sequence detection
- Software reset ($\overline{\text{SRST}}$)
- Two independent, 32-entry FIFOs for transmit and receive
- The peripheral clock can be totally asynchronous with the module clock. The module clock determines baud rate. This allows frequency scaling on peripheral clock (such as during DVFS mode) while remaining the module clock frequency and baud rate.

59.1.2 Modes of Operation

The UART uses the following modes of operation:

- Serial RS-232 NRZ format
- IrDA

59.2 External Signals

Table 59-1 lists the conventions for representing signals.

Table 59-1. Module Signal Conventions

Category	Convention	Example(s)
Off-chip signal	Uppercase (all capital letters)	TXD
Internal signal[1]	Lowercase italics	*core_int*
Active low signal	_B (_b) suffix or overbar	RESET_EN_B or $\overline{\text{RESET\_EN}}$
Range of bussed or commonly named signals	Beginning and end points of the range are: • Separated by a colon. • Surrounded by square brackets.	ADDR[31:0] CSE_B[7:0] or $\overline{\text{CSE}}$[7:0]
Individual signal in a range of bussed or commonly named signals	Individual number in the range appears without a colon or square brackets	ADDR31 CSE0_B or $\overline{\text{CSE0}}$

1 Internal signals are for reference only in descriptions of internal module or SoC functionality.

59.2.1 Detailed Signal Descriptions

Table 59-2 describes all UART signals that connect off-chip.

Table 59-2. Off-Chip Module Signals

Signal name	I/O	Active state	Description	Reset state
Serial/IrDA Signals				
RXD	I		Serial / infrared data receive	—
TXD	O		Serial/infrared data transmit	High
Modem Control Signals				
$\overline{\text{CTS}}$	O	Low	Clear to send	High
$\overline{\text{RTS}}$	I	Low	Request to send	
$\overline{\text{DSR}}$	I/O	Low	Data set ready	High
$\overline{\text{DCD}}$	I/O	Low	Data carrier detected	High
$\overline{\text{DTR}}$	I/O	Low	Data terminal ready	
$\overline{\text{RI}}$	I/O	Low	Ring indicator	High
Interrupts				
$\overline{\textit{interrupt\_uart}}$	O	Low	UART interrupt	High
DMA Requests				
$\overline{\textit{dma\_req\_rx}}$	O	Low	Receiver DMA request	High
$\overline{\textit{dma\_req\_tx}}$	O	Low	Transmitter DMA request	High

Table 59-2. Off-Chip Module Signals (continued)

Signal name	I/O	Active state	Description	Reset state
Clocks				
peripheral_clock	I		Peripheral clock	—
module_clock	I		Clock source for the module's logic	—
Special Signals				
stop_req	I	High	Module stop mode	—
doze_req	I	High	Module doze mode	—

59.2.1.1 Serial/IrDA Signals

The serial/IrDA signals are as follows:

- RXD (Data Receive)—Input asynchronous data receive in Serial and IrDA modes.
- TXD (Data Transmit)—Output asynchronous data transmit in Serial and IrDA modes.

59.2.1.2 Modem Control Signals

The modem control signals are as follows:

- $\overline{\text{CTS}}$ (Clear To Send)
  - — Output in DCE and DTE mode
  - — Informs the remote modem that UART is ready to receive data
- $\overline{\text{RTS}}$ (Request To Send)
  - — Input in DCE and DTE mode
  - — Informs UART that remote modem is ready to receive data
- $\overline{\text{DSR}}$ (Data Set Ready)
  - — Input in DTE mode. Indicates to UART that remote modem is operational.
  - — Output in DCE mode. Indicates to remote modem that UART is operational.
- $\overline{\text{DCD}}$ (Data Carrier Detected)
  - — Input in DTE mode. Indicates to UART that a good carrier is being received from the remote modem.
  - — Output in DCE mode. Indicates to remote device that a good carrier is being received from the UART.
- $\overline{\text{DTR}}$ (Data Terminal Ready)
  - — Input in DCE mode. Indicates to UART (in DCE mode) that remote device (in DTE mode) is operational.
  - — Output in DTE mode. Indicates to remote modem (in DCE mode) that UART (in DTE mode) is operational.

- $\overline{RI}$ (Ring Indicator)
  - Input in DTE mode. Indicates to UART that remote modem is detecting a ringing tone.
  - Output in DCE mode. Indicates to remote device that UART is detecting a ringing tone.

59.2.1.3 Interrupt Signals

UART Interrupt, $\overline{interrupt\_uart}$, is the interrupt signal for the output interrupt request.

59.2.1.4 DMA Request Signals

The DMA request signals are as follows:

- $\overline{dma\_req\_rx}$ (Receiver DMA Request)—Output DMA Request signal for receiver interface.
- $\overline{dma\_req\_tx}$ (Transmitter DMA Request)—Output DMA Request signal for transmitter interface. Set at 0 when TXDMAEN (UCR1[3]) is at 1 and TRDY (USR1[13]) is also at 1.

59.2.1.5 Clock Signals

The clock signals are as follows:

- *peripheral_clock* (Peripheral Clock)—See Section 59.4.2, Clocks" for more information about *peripheral_clock*.
- *module_clock* (Module Clock)—See Section 59.4.2, Clocks" for more information about *module_clock*.

59.2.1.6 Special Signals

Special signals are as follows:

- *stop_req* (Stop Mode)
  - Input stop mode
  - Indicates UART that MCU is going to enter in Stop Mode and clocks are going to stop running
  - See Section 59.4.8, Low Power Modes" for more information about Stop Mode.
- *doze_req* (Doze Mode)
  - Input doze mode
  - MCU requests UART to switch in doze mode (power saving mode)
  - See Section 59.4.8, Low Power Modes" for more information about Doze Mode.

59.3 Memory Map and Register Definition

This section includes the module memory map and detailed descriptions of all registers.

59.3.1 Memory Map

Table 59-3 is the UART memory map.

Table 59-3. UART Memory Map

Offset Address (Register Abbreviation)	Register	Access	Reset Value	Section/Page
0x0000 (URXD)	UART Receiver Register	R	00– –	59.3.3.1/59-10
0x0004–0x003c	Reserved	—	—	—
0x0040 (UTXD)	UART Transmitter Register	W	00– –	59.3.3.2/59-12
0x0044–0x007c	Reserved	—	—	—
0x0080 (UCR1)	UART Control Register 1	R/W	0000	59.3.3.3/59-13
0x0084 (UCR2)	UART Control Register 2	R/W	0001	59.3.3.4/59-15
0x0088 (UCR3)	UART Control Register 3	R/W	0700	59.3.3.5/59-17
0x008c (UCR4)	UART Control Register 4	R/W	8000	59.3.3.6/59-19
0x0090 (UFCR)	UART FIFO Control Register	R/W	0801	59.3.3.7/59-21
0x0094 (USR1)	UART Status Register 1	R/W	2040	59.3.3.8/59-22
0x0098 (USR2)	UART Status Register 2	R/W	4028	59.3.3.9/59-24
0x009c (UESC)	UART Escape Character Register	R/W	002b	59.3.3.10/59-27
0x00a0 (UTIM)	UART Escape Timer Register	R/W	0000	59.3.3.11/59-27
0x00a4 (UBIR)	UART BRM Incremental Register	R/W	0000	59.3.3.12/59-28
0x00a8 (UBMR)	UART BRM Modulator Register	R/W	0000	59.3.3.13/59-29
0x00ac (UBRC)	UART Baud Rate Count Register	R	0004	59.3.3.14/59-30
0x00b0 (ONEMS)	UART One Millisecond Register	R/W	000000	59.3.3.15/59-31
0x00b4 (UTS)	UART Test Register	R/W	0060	59.3.3.16/59-32

59.3.2 Register Summary

Table 59-5 is the register summary table.

The UART supports 8-bit, 16-bit and 32-bit accesses to 32-bit memory-mapped addresses. Any access to unmapped memory location yields a transfer error.

All the memory mapped registers are 32 bits wide. However, the 16 MSB are not used for all of the registers except the ONEMS register:

- For 32-bits write access, the 16 MSB is not taken into account for all of the registers except the ONEMS.
- For 32-bits read access the 16 MSB is read as 0 for all of the registers except the ONEMS.

The ONEMS register is expanded from 16 bits to 24 bits in order to support the high frequency of the BRM internal clock *ref_clk* (*module_clock* after divider). The ONEMS register can be accessed in the way of 8-bits, 16-bits or 32-bits.

- For 32-bits write access, the 8 MSB of the ONEMS will not be taken into account.
- For 32-bits read access, the 8 MSB of the ONEMS will be read as 0.

All registers except the ONEMS described in this section are for 16 LSB. The ONEMS register is for 24 LSB.

Figure 59-2 shows the key to the register fields and Table 59-4 shows the register figure conventions.

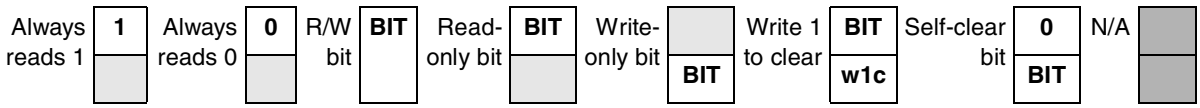


Figure 59-2. Key to Register Fields

Table 59-4. Register Figure Convention

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
slfclr	Self-clearing bit. Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 59-5 shows the UART register summary.

Table 59-5. UART Register Summary

Offset Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0000 (URXD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	CHARRDY	ERR	OVRRUN	FRMERR	BRK	PRERR	0	0	RX_DATA							
	W																
0x0040 (UTXD)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0								
	W									TX_DATA							
0x0080 (UCR1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R/W	ADEN	ADBR	TRDYEN	IDEN	ICD		RRDYEN	RXDMAEN	IREN	TXMPTYEN	RTSDEN	SNDBRK	TXDMAEN	ATDMAEN	DOZE	UARTEN
0x0084 (UCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R/W	ESCI	IRTS	CTSC	CTS	ESCEN	RTEC		PREN	PROE	STPB	WS	RTSEN	ATEN	TXEN	RXEN	SRST
0x0088 (UCR3)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R/W	DPEC		DTREN	PARERREN	FRAERREN	DSR	DCD	RI	ADNIMP	RXDSEN	AIRINTEN	AWAKEN	DTRDEN	RXDMUXSEL	INVT	ACIEN
0x008c (UCR4)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R/W	CTSTL						INVR	ENIRI	WKEN	IDDMAEN	IRSC	LPBYP	BKEN	BKEN	OREN	DREN

Table 59-5. UART Register Summary (continued)

Offset Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0090 (UFCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	TXTL						RFDIV			DCEDTE	RXTL					
	W																
0x0094 (USR1)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	PARITYERR	RTSS	TRDY	RTSD	ESCF	FRAMERR	RRDY	AGTIM	DTRD	RXDS	AIRINT	AWAKE	0	0	0	0
	W	w1c			w1c	w1c	w1c		w1c	w1c		w1c	w1c				
0x0098 (USR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	ADET	TXFE	DTRF	IDLE	ACST	RIDELT	RIIN	IRINT	WAKE	DCDDELT	DCDIN	RTSF	TXDC	BRCD	ORE	RDR
	W	w1c		w1c	w1c	w1c	w1c		w1c	w1c	w1c		w1c		w1c	w1c	
0x009c (UESC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	ESC_CHAR							
	W																
0x00a0 (UTIM)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	TIM											
	W																
0x00a4 (UBIR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	INC															
	W																
0x00a8 (UBMR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	MOD															
	W																

Table 59-5. UART Register Summary (continued)

Offset Address (Name Abbreviation)		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00ac (UBRC)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	BCNT															
	W																
0x00b0 (ONEMS)	R	0	0	0	0	0	0	0	0								
	W																
	R	ONEMS															
	W																
0x00b4 (UTS)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	FRCPERR	LOOP	DBGEN	LOOPIR	RXDBG	0	0	TXEMPTY	RXEMPTY	TXFULL	RXFULL	0	0	SOFTRST
	W																

59.3.3 Register Descriptions

This section provides detailed descriptions of the UART registers.

59.3.3.1 UART Receiver Register (URXD)

See Figure 59-3 for illustration of valid bits in the UART Receiver Register and Table 59-6 for description of the bit fields.

0x0000 (URXD) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHARRDY	ERR	OVRRUN	FRMERR	BRK	PRERR	0	0	RX_DATA							
W																
RESET	0	0	0	0	0	0	0	0	—	—	—	—	—	—	—	—

Figure 59-3. UART Receiver Register

Table 59-6. UART Receiver Register Field Descriptions

Field	Description
31–16	Reserved
15 CHARRDY	Character Ready. This read-only bit indicates an invalid read when the FIFO becomes empty and software tries to read the same old data. This bit should not be used for polling for data written to the RX FIFO. 0 Character in RX_DATA field and associated flags are invalid. 1 Character in RX_DATA field and associated flags valid and ready for reading.
14 ERR	Error Detect. Indicates whether the character present in the RX_DATA field has an error (OVRRUN, FRMERR, BRK or PRERR) status. The ERR bit is updated and valid for each received character. 0 No error status was detected 1 An error status was detected
13 OVRRUN	Receiver Overrun. This read-only bit, when HIGH, indicates that the corresponding character was stored in the last position (32nd) of the Rx FIFO. Even if a 33rd character has not been detected, this bit will be set to ‘1’ for the 32nd character. 0 No RxFIFO overrun was detected 1 A RxFIFO overrun was detected
12 FRMERR	Frame Error. Indicates whether the current character had a framing error (a missing stop bit) and is possibly corrupted. FRMERR is updated for each character read from the RxFIFO. 0 The current character has no framing error 1 The current character has a framing error
11 BRK	BREAK Detect. Indicates whether the current character was detected as a BREAK character. The data bits and the stop bit are all 0. The FRMERR bit is set when BRK is set. When odd parity is selected, PRERR is also set when BRK is set. BRK is valid for each character read from the RxFIFO. 0 The current character is not a BREAK character 1 The current character is a BREAK character
10 PRERR	Parity Error. Indicates whether the current character was detected with a parity error and is possibly corrupted. PRERR is updated for each character read from the RxFIFO. When parity is disabled, PRERR always reads as 0. 0 = No parity error was detected for data in the RX_DATA field 1 = A parity error was detected for data in the RX_DATA field
9–8	Reserved
7–0 RX_DATA	Received Data. Holds the received character. In 7-bit mode, the most significant bit (MSB) is forced to 0. In 8-bit mode, all bits are active.

NOTE

The UART will yield a transfer error on the peripheral bus when core is reading URXD register with receive interface disabled (RXEN = 0 or UARTEN = 0).

59.3.3.2 UART Transmitter Register (UTXD)

See Figure 59-4 for illustration of valid bits in the UART Transmitter Register and Table 59-7 for description of the bit fields.

0x0040 (UTXD) Access: User write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0								
W									TX_DATA							
RESET	0	0	0	0	0	0	0	0	—	—	—	—	—	—	—	—

Figure 59-4. UART Transmitter Register

Table 59-7. UART Transmitter Register Field Descriptions

Field	Description
31–16	Reserved
15–8	Reserved
7–0 TX_DATA	Transmit Data. Holds the parallel transmit data inputs. In 7-bit mode, D7 is ignored. In 8-bit mode, all bits are used. Data is transmitted least significant bit (LSB) first. A new character is transmitted when the TX_DATA field is written. The TX_DATA field must be written only when the TRDY bit is high to ensure that corrupted data is not sent.

NOTE

The UART will yield a transfer error on the peripheral bus when core is writing into URXD register with transmit interface disabled (TXEN=0 or UARTEN=0).

Memory space between URXD and UTXD registers is reserved. Any read or write access to this space will be considered as an invalid access and yield a transfer error.

59.3.3.3 UART Control Register 1 (UCR1)

See Figure 59-5 for illustration of valid bits in the UART Control Register 1 and Table 59-8 for description of the bit fields.

0x0080 (UCR1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADEN	ADBR	TRDY EN	IDEN	ICD		RRDY EN	RXD MAE N	IREN	TXMP TYEN	RTSD EN	SNDB RK	TXD MAE N	ATDM AEN	DOZE	UAR TEN
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 59-5. UART Control Register 1

Table 59-8. UART Control Register 1 Field Descriptions

Field	Description
31–16	Reserved
15 ADEN	Automatic Baud Rate Detection Interrupt Enable. Enables/Disables the automatic baud rate detect complete (ADET) bit to generate an interrupt ($\overline{interrupt\_uart}$ = 0). 0 Disable the automatic baud rate detection interrupt 1 Enable the automatic baud rate detection interrupt
14 ADBR	Automatic Detection of Baud Rate. Enables/Disables automatic baud rate detection. When the ADBR bit is set and the ADET bit is cleared, the receiver detects the incoming baud rate automatically. The ADET flag is set when the receiver verifies that the incoming baud rate is detected properly by detecting an ASCII character “A” or “a” (0x61 or 0x41). 0 Disable automatic detection of baud rate 1 Enable automatic detection of baud rate
13 TRDYEN	Transmitter Ready Interrupt Enable. Enables/Disables the transmitter Ready Interrupt (TRDY) when the transmitter has one or more slots available in the TxFIFO. The fill level in the TXFIFO at which an interrupt is generated is controlled by TxTL bits. When TRDYEN is negated, the transmitter ready interrupt is disabled. **Note:** An interrupt will be issued as long as TRDYEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TRDY interrupt. 0 Disable the transmitter ready interrupt 1 Enable the transmitter ready interrupt
12 IDEN	Idle Condition Detected Interrupt Enable. Enables/Disables the IDLE bit to generate an interrupt ($\overline{interrupt\_uart}$ = 0). 0 Disable the IDLE interrupt 1 Enable the IDLE interrupt

Table 59-8. UART Control Register 1 Field Descriptions (continued)

Field	Description
11–10 ICD	Idle Condition Detect. Controls the number of frames RXD is allowed to be idle before an idle condition is reported. 00 Idle for more than 4 frames 01 Idle for more than 8 frames 10 Idle for more than 16 frames 11 Idle for more than 32 frames
9 RRDYEN	Receiver Ready Interrupt Enable. Enables/Disables the RRDY interrupt when the RxFIFO contains data. The fill level in the RxFIFO at which an interrupt is generated is controlled by the RXTL bits. When RRDYEN is negated, the receiver ready interrupt is disabled. 0 Disables the RRDY interrupt 1 Enables the RRDY interrupt
8 RXDMAEN	Receive Ready DMA Enable. Enables/Disables the receive DMA request $\overline{dma\_req\_rx}$ when the receiver has data in the RxFIFO. The fill level in the RxFIFO at which a DMA request is generated is controlled by the RXTL bits. When negated, the receive DMA request is disabled. 0 Disable DMA request 1 Enable DMA request
7 IREN	Infrared Interface Enable. Enables/Disables the IR interface. Refer to the IR interface description in Section 59.4.7, Infrared Interface," for more information. 0 Disable the IR interface 1 Enable the IR interface
6 TXMPTYEN	Transmitter Empty Interrupt Enable. Enables/Disables the transmitter FIFO empty (TXFE) interrupt. $\overline{interrupt\_uart}$. When negated, the TXFE interrupt is disabled. **Note:** An interrupt will be issued as long as TXMPTYEN and TXFE are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXFE interrupt. 0 Disable the transmitter FIFO empty interrupt 1 Enable the transmitter FIFO empty interrupt
5 RTSDEN	RTS Delta Interrupt Enable. Enables/Disables the RTSD interrupt. The current status of the $\overline{RTS}$ pin is read in the RTSS bit. 0 Disable RTSD interrupt 1 Enable RTSD interrupt
4 SNDBRK	Send BREAK. Forces the transmitter to send a BREAK character. The transmitter finishes sending the character in progress (if any) and sends BREAK characters until SNDBRK is reset. Because the transmitter samples SNDBRK after every bit is transmitted, it is important that SNDBRK is asserted high for a sufficient period of time to generate a valid BREAK. After the BREAK transmission completes, the UART transmits 2 mark bits. The user can continue to fill the TxFIFO and any characters remaining are transmitted when the BREAK is terminated. 0 Do not send a BREAK character 1 Send a BREAK character (continuous 0s)
3 TXDMAEN	Transmitter Ready DMA Enable. Enables/Disables the transmit DMA request $\overline{dma\_req\_tx}$ when the transmitter has one or more slots available in the TxFIFO. The fill level in the TxFIFO that generates the $\overline{dma\_req\_tx}$ is controlled by the TXTL bits. **Note:** A DMA request will be issued as long as TXDMAEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the transmit DMA request. 0 Disable transmit DMA request 1 Enable transmit DMA request

Table 59-8. UART Control Register 1 Field Descriptions (continued)

Field	Description
2 ATDMAEN	Aging DMA Timer Enable. Enables/Disables the receive DMA request *dma_req_rx* for the aging timer interrupt (triggered with AGTIM flag in USR1[8]). 0 Disable AGTIM DMA request 1 Enable AGTIM DMA request
1 DOZE	DOZE. Determines the UART enable condition in the DOZE state. When *doze_req* input pin is at '1', (the CPU executes a doze instruction and the system is placed in the Doze State), the DOZE bit affects operation of the UART. While in the Doze State, if this bit is asserted, the UART is disabled. Refer to the description in Section 59.4.8, Low Power Modes." 0 The UART is enabled when in DOZE state 1 The UART is disabled when in DOZE state
0 UARTEN	UART Enable. Enables/Disables the UART. If UARTEN is negated in the middle of a transmission, the transmitter stops and pulls the TXD line to a logic 1. UARTEN must be set to 1 before any access to UTXD and URXD registers, otherwise a transfer error is returned. This bit can be set to 1 along with other bits in this register. There is no restriction to the sequence of programing this bit and other control registers. 0 Disable the UART 1 Enable the UART

59.3.3.4 UART Control Register 2 (UCR2)

See Figure 59-6 for illustration of valid bits in the UART Control Register 2 and Table 59-9 for description of the bit fields.

0x0084 (UCR2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ESCI	IRTS	CTSC	CTS	ESCEN	RTEC		PREN	PROE	STPB	WS	RTSEN	ATEN	TXEN	RXEN	SRST
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 59-6. UART Control Register 2

Table 59-9. UART Control Register 2 Field Descriptions

Name	Description
31–16	Reserved
15 ESCI	Escape Sequence Interrupt Enable. Enables/Disables the ESCF bit to generate an interrupt. 0 Disable the escape sequence interrupt 1 Enable the escape sequence interrupt

Table 59-9. UART Control Register 2 Field Descriptions (continued)

Name	Description
14 IRTS	Ignore RTS Pin. Forces the RTS input signal presented to the transmitter to always be asserted (set to low), effectively ignoring the external pin. When in this mode, the RTS pin serves as a general purpose input. 0 Transmit only when the RTS pin is asserted 1 Ignore the RTS pin
13 CTSC	CTS Pin Control. Controls the operation of the $\overline{CTS}$ output pin. When CTSC is asserted, the $\overline{CTS}$ output pin is controlled by the receiver. When the RxFIFO is filled to the level of the programmed trigger level and the start bit of the overflowing character (TRIGGER LEVEL + 1) is validated, the $\overline{CTS}$ output pin is negated to indicate to the far-end transmitter to stop transmitting. When the trigger level is programmed for less than 32, the receiver continues to receive data until the RxFIFO is full. When the CTSC bit is negated, the $\overline{CTS}$ output pin is controlled by the CTS bit. On reset, because CTSC is cleared to 0, the $\overline{CTS}$ pin is controlled by the CTS bit, which again is cleared to 0 on reset. This means that on reset the $\overline{CTS}$ signal is negated. 0 The $\overline{CTS}$ pin is controlled by the CTS bit 1 The $\overline{CTS}$ pin is controlled by the receiver
12 CTS	Clear to Send. Controls the $\overline{CTS}$ pin when the CTSC bit is negated. CTS has no function when CTSC is asserted. 0 The $\overline{CTS}$ pin is high (inactive) 1 The $\overline{CTS}$ pin is low (active)
11 ESCEN	Escape Enable. Enables/Disables the escape sequence detection logic. 0 Disable escape sequence detection 1 Enable escape sequence detection
10-9 RTEC	Request to Send Edge Control. Selects the edge that triggers the RTS interrupt. This has no effect on the RTS delta interrupt. RTEC has an effect only when RTSEN = 1 (see Table 59-23). 00 Trigger interrupt on a rising edge 01 Trigger interrupt on a falling edge 1X Trigger interrupt on any edge
8 PREN	Parity Enable. Enables/Disables the parity generator in the transmitter and parity checker in the receiver. When PREN is asserted, the parity generator and checker are enabled, and disabled when PREN is negated. 0 Disable parity generator and checker 1 Enable parity generator and checker
7 PROE	Parity Odd/Even. Controls the sense of the parity generator and checker. When PROE is high, odd parity is generated and expected. When PROE is low, even parity is generated and expected. PROE has no function if PREN is low. 0 Even parity 1 Odd parity
6 STPB	Stop. Controls the number of stop bits after a character. When STPB is low, 1 stop bit is sent. When STPB is high, 2 stop bits are sent. STPB also affects the receiver. 0 The transmitter sends 1 stop bit. The receiver expects 1 or more stop bits. 1 The transmitter sends 2 stop bits. The receiver expects 2 or more stop bits.
5 WS	Word Size. Controls the character length. When WS is high, the transmitter and receiver are in 8-bit mode. When WS is low, they are in 7-bit mode. The transmitter ignores bit 7 and the receiver sets bit 7 to 0. WS can be changed in-between transmission (reception) of characters, however not when a transmission (reception) is in progress, in which case the length of the current character being transmitted (received) is unpredictable. 0 7-bit transmit and receive character length (not including START, STOP or PARITY bits) 1 8-bit transmit and receive character length (not including START, STOP or PARITY bits)

Table 59-9. UART Control Register 2 Field Descriptions (continued)

Name	Description
4 RTSEN	Request to Send Interrupt Enable. Controls the RTS edge sensitive interrupt. When RTSEN is asserted and the programmed edge is detected on the $\overline{RTS}$ pin (the RTSF bit is asserted), an interrupt will be generated on the *interrupt_uart* pin. (see Table 59-23). 0 Disable request to send interrupt 1 Enable request to send interrupt
3 ATEN	Aging Timer Enable. This bit is used to enable the aging timer interrupt (triggered with AGTIM) 0 AGTIM interrupt disabled 1 AGTIM interrupt enabled
2 TXEN	Transmitter Enable. Enables/Disables the transmitter. When TXEN is negated the transmitter is disabled and idle. When the UARTEN and TXEN bits are set the transmitter is enabled. If TXEN is negated in the middle of a transmission, the UART disables the transmitter immediately, and starts marking 1s. The transmitter FIFO cannot be written when this bit is cleared. 0 Disable the transmitter 1 Enable the transmitter
1 RXEN	Receiver Enable. Enables/Disables the receiver. When the receiver is enabled, if the RXD input is already low, the receiver does not recognize BREAK characters, because it requires a valid 1-to-0 transition before it can accept any character. 0 Disable the receiver 1 Enable the receiver
0 $\overline{SRST}$	Software Reset. Once the software writes 0 to $\overline{SRST}$, the software reset remains active for 4 *module_clock* cycles before the hardware deasserts $\overline{SRST}$. The software can only write 0 to $\overline{SRST}$. Writing 1 to $\overline{SRST}$ is ignored. 0 Reset the transmit and receive state machines, all FIFOs and register USR1, USR2, UBIR, UBMR, UBRC , URXD, UTXD and UTS[6–3]. 1 No reset

59.3.3.5 UART Control Register 3 (UCR3)

See Figure 59-7 for illustration of valid bits in the UART Control Register 3 and Table 59-10 for description of the bit fields.

0x0088 (UCR3) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DPEC		DTREN	PARERREN	FRAERREN	DSR	DCD	RI	ADNIMP	RXDSEN	AIRINTEN	AWAKEN	DTRDEN	RXDMUXSEL	INVT	ACIEN
RESET	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

Figure 59-7. UART Control Register 3

Table 59-10. UART Control Register 3 Field Descriptions

Name	Description
31–16	Reserved
15–14 DPEC	DTR/DSR Interrupt Edge Control. These bits control the edge of $\overline{\text{DTR}}$ (DCE) or $\overline{\text{DSR}}$ (DTE) on which an interrupt will be generated. An interrupt will only be generated if the DTREN bit is set. 00 interrupt generated on rising edge 01 interrupt generated on falling edge 1X interrupt generated on either edge
13 DTREN	Data Terminal Ready Interrupt Enable. When this bit is set, it will enable the status bit DTRF (USR2 [13]) (DTR/DSR edge sensitive interrupt) to cause an interrupt. 0 Data Terminal Ready Interrupt Disabled 1 Data Terminal Ready Interrupt Enabled
12 PARERREN	Parity Error Interrupt Enable. Enables/Disables the interrupt. When asserted, PARERREN causes the PARITYERR bit to generate an interrupt. 0 Disable the parity error interrupt 1 Enable the parity error interrupt
11 FRAERREN	Frame Error Interrupt Enable. Enables/Disables the interrupt. When asserted, FRAERREN causes the FRAMERR bit to generate an interrupt. 0 Disable the frame error interrupt 1 Enable the frame error interrupt
10 DSR	Data Set Ready. This bit is used by software to control the DSR/DTR output pin for the modem interface. In DCE mode it applies to $\overline{\text{DSR}}$ and in DTE mode it applies to $\overline{\text{DTR}}$. 0 DSR/ DTR pin is logic zero 1 DSR/ DTR pin is logic one
9 DCD	Data Carrier Detect. In DCE mode this bit is used by software to control the $\overline{\text{DCD}}$ output pin for the modem interface. In DTE mode, when this bit is set, it will enable the status bit DCDDELT (USR2 (6)) to cause an interrupt. 0 $\overline{\text{DCD}}$ pin is logic zero (DCE mode) 1 $\overline{\text{DCD}}$ pin is logic one (DCE mode) 0 DCDDELT interrupt disabled (DTE mode) 1 DCDDELT interrupt enabled (DTE mode)
8 RI	Ring Indicator. In DCE mode this bit is used by software to control the $\overline{\text{RI}}$ output pin for the modem interface. In DTE mode, when this bit is set, it will enable the status bit RIDELT (USR2 (10)) to cause an interrupt. 0 $\overline{\text{RI}}$ pin is logic zero (DCE mode) 1 $\overline{\text{RI}}$ pin is logic one (DCE mode) 0 RIDELT interrupt disabled (DTE mode) 1 RIDELT interrupt enabled (DTE mode)
7 ADNIMP	Autobaud Detection Not Improved—. Disables new features of autobaud detection (refer to Section 59.4.5.6.2, Baud Rate Automatic Detection Protocol Improved," for more details). 0 Autobaud detection new features selected 1 Keep old autobaud detection mechanism
6 RXDSEN	Receive Status Interrupt Enable. Controls the receive status interrupt ($\overline{\textit{interrupt\_uart}}$). When this bit is enabled and RXDS status bit is set, the interrupt $\overline{\textit{interrupt\_uart}}$ will be generated. 0 Disable the RXDS interrupt 1 Enable the RXDS interrupt
5 AIRINTEN	Asynchronous IR WAKE Interrupt Enable. Controls the asynchronous IR WAKE interrupt. An interrupt is generated when AIRINTEN is asserted and a pulse is detected on the RXD pin. 0 Disable the AIRINT interrupt 1 Enable the AIRINT interrupt

Table 59-10. UART Control Register 3 Field Descriptions (continued)

Name	Description
4 AWAKEN	Asynchronous WAKE Interrupt Enable. Controls the asynchronous WAKE interrupt. An interrupt is generated when AWAKEN is asserted and a falling edge is detected on the RXD pin. 0 Disable the AWAKE interrupt 1 Enable the AWAKE interrupt
3 DTRDEN	Data Terminal Ready Delta Enable. Enables / Disables the asynchronous DTRD interrupt. When DTRDEN is asserted and an edge (rising or falling) is detected on $\overline{\text{DTR}}$ (in DCE mode) or on $\overline{\text{DSR}}$ (in DTE mode), then an interrupt is generated. 0 Disable DTRD interrupt 1 Enable DTRD interrupt
2 RXDMUXSEL	RXD Muxed Input Selected. Selects proper input pins for serial and Infrared input signal. **Note:** In this chip, UARTs are used in MUXED mode, so that this bit should always be set. 0 Input pin RXD is not used for serial and IR interfaces 1 Input pin RXD is used for serial and IR interfaces
1 INVT	Inverted Infrared Transmission. Sets the active level for the transmission. When INVT is cleared, the infrared logic block transmits a positive IR 3/16 pulse for all 0s and 0s are transmitted for 1s. When INVT is set (INVT = 1), the infrared logic block transmits an active low or negative infrared 3/16 pulse for all 0s and 1s are transmitted for 1s. 0 Active low transmission 1 Active high transmission.
0 ACIEN	Autobaud Counter Interrupt Enable. This bit is used to enable the autobaud counter stopped interrupt (triggered with ACST (USR2[11]). 0 ACST interrupt disabled 1 ACST interrupt enabled

59.3.3.6 UART Control Register 4 (UCR4)

See Figure 59-8 for illustration of valid bits in the UART Control Register 4 and Table 59-11 for description of the bit fields.

0x008c (UCR4) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CTSTL						INVR	ENIRI	WKEN	IDDMAEN	IRSC	LPBYP	TCEN	BKEN	OREN	DREN
RESET	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 59-8. UART Control Register 4

Table 59-11. UART Control Register 4 Field Descriptions

Name	Description
31–16	Reserved
15–10 CTSTL	CTS Trigger Level. Controls the threshold at which the $\overline{CTS}$ pin is deasserted by the RxFIFO. After the trigger level is reached and the $\overline{CTS}$ pin is deasserted, the RxFIFO continues to receive data until it is full. The CTSTL bits are encoded as shown in the Settings column. 000000 0 characters received 000001 1 characters in the RxFIFO ... ... 100000 32 characters in the RxFIFO (maximum) All Other Settings Reserved
9 INVR	Inverted Infrared Reception. Determines the logic level for the detection. When cleared, the infrared logic block expects an active low or negative IR 3/16 pulse for 0s and 1s are expected for 1s. When INVR is set (INVR1), the infrared logic block expects an active high or positive IR 3/16 pulse for 0s and 0s are expected for 1s. 0 Active low detection 1 Active high detection
8 ENIRI	Serial Infrared Interrupt Enable. Enables/Disables the serial infrared interrupt. 0 Serial infrared Interrupt disabled 1 Serial infrared Interrupt enabled
7 WKEN	WAKE Interrupt Enable. Enables/Disables the WAKE bit to generate an interrupt. The WAKE bit is set at the detection of a start bit by the receiver. 0 Disable the WAKE interrupt 1 Enable the WAKE interrupt
6 IDDMAEN	DMA IDLE Condition Detected Interrupt Enable Enables/Disables the receive DMA request $\overline{dma\_req\_rx}$ for the IDLE interrupt (triggered with IDLE flag in USR2[12]). 0 DMA IDLE interrupt disabled 1 DMA IDLE interrupt enabled
5 IRSC	IR Special Case. Selects the clock for the vote logic. When set, IRSC switches the vote logic clock from the sampling clock to the UART reference clock. The IR pulses are counted a predetermined amount of time depending on the reference frequency. Refer to Section 59.4.7.3, InfraRed Special Case (IRSC) Bit. 0 The vote logic uses the sampling clock (16x baud rate) for normal operation 1 The vote logic uses the UART reference clock
4 LPBYP	Low Power Bypass. Allows to bypass the low power new features in UART. To use during debug phase. 0 Low power features enabled 1 Low power features disabled
3 TCEN	Transmit Complete Interrupt Enable. Enables/Disables the TXDC bit to generate an interrupt ($\overline{interrupt\_uart}$ = 0) **Note:** An interrupt will be issued as long as TCEN and TXDC are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXDC interrupt. 0 Disable TXDC interrupt 1 Enable TXDC interrupt
2 BKEN	BREAK Condition Detected Interrupt Enable. Enables/Disables the BRCD bit to generate an interrupt. 0 Disable the BRCD interrupt 1 Enable the BRCD interrupt

Table 59-11. UART Control Register 4 Field Descriptions (continued)

Name	Description
1 OREN	Receiver Overrun Interrupt Enable. Enables/Disables the ORE bit to generate an interrupt. 0 Disable ORE interrupt 1 Enable ORE interrupt
0 DREN	Receive Data Ready Interrupt Enable. Enables/Disables the RDR bit to generate an interrupt. 0 Disable RDR interrupt 1 Enable RDR interrupt

59.3.3.7 UART FIFO Control Register (UFCR)

See Figure 59-9 for illustration of valid bits in the UART FIFO Control Register and Table 59-12 for description of the bit fields.

0x0090 (UFCR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXTL						RFDIV			DCE DTE	RXTL					
RESET	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

Figure 59-9. UART FIFO Control Register

Table 59-12. UART FIFO Control Register Field Descriptions

Name	Description
31–16	Reserved
15–10 TXTL	Transmitter Trigger Level. Controls the threshold at which a maskable interrupt is generated by the TxFIFO. A maskable interrupt is generated whenever the data level in the TxFIFO falls below the selected threshold. The bits are encoded as shown in the Settings column. 000000 Reserved 000001 Reserved 000010 TxFIFO has 2 or fewer characters ... ... 011111 TxFIFO has 31 or fewer characters 100000 TxFIFO has 32 characters (maximum) All Other Settings Reserved

Table 59-12. UART FIFO Control Register Field Descriptions (continued)

Name	Description
9–7 RFDIV	Reference Frequency Divider. Controls the divide ratio for the reference clock. The input clock is *module_clock*. The output from the divider is *ref_clk* which is used by BRM to create the 16x baud rate oversampling clock (*brm_clk*). 000 Divide input clock by 6 001 Divide input clock by 5 010 Divide input clock by 4 011 Divide input clock by 3 100 Divide input clock by 2 101 Divide input clock by 1 110 Divide input clock by 7 111 Reserved
6 DCEDTE	DCE/DTE mode select. Selects data communication equipment (DCE) or data terminal equipment (DTE) mode. This bit controls the pin direction of the bidirectional modem pins DSR, DCD, DTR and RI. 0 DCE mode selected 1 DTE mode selected
5–0 RXTL	Receiver Trigger Level. Controls the threshold at which a maskable interrupt is generated by the RxFIFO. A maskable interrupt is generated whenever the data level in the RxFIFO reaches the selected threshold. The RXTL bits are encoded as shown in the Settings column. 000000 0 characters received 000001 RxFIFO has 1 character ... ... 011111 RxFIFO has 31 characters 100000 RxFIFO has 32 characters (maximum) All Other Settings Reserved

59.3.3.8 UART Status Register 1 (USR1)

See Figure 59-10 for illustration of valid bits in the UART Status Register 1 and Table 59-13 for description of the bit fields.

0x0094 (USR1) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PARITYERR	RTSS	TRDY	RTSD	ESCF	FRAMERR	RRDY	AGTIM	DTRD	RXDS	AIRINT	AWAKE	0	0	0	0
W	w1c			w1c	w1c	w1c		w1c	w1c		w1c	w1c				
RESET	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0

Figure 59-10. UART Status Register 1

Table 59-13. UART Status Register 1 Field Descriptions

Name	Description
31–16	Reserved
15 PARITYERR	Parity Error Interrupt Flag. Indicates a parity error is detected. PARITYERR is cleared by writing 1 to it. Writing 0 to PARITYERR has no effect. When parity is disabled, PARITYERR always reads 0. At reset, PARITYERR is set to 0. 0 No parity error detected 1 Parity error detected (write 1 to clear)
14 RTSS	$\overline{\text{RTS}}$ Pin Status. Indicates the current status of the $\overline{\text{RTS}}$ pin. A “snapshot” of the pin is taken immediately before RTSS is presented to the data bus. RTSS cannot be cleared because all writes to RTSS are ignored. At reset, RTSS is set to 0. 0 The $\overline{\text{RTS}}$ pin is high (inactive) 1 The $\overline{\text{RTS}}$ pin is low (active)
13 TRDY	Transmitter Ready Interrupt / DMA Flag. Indicates that the TxFIFO emptied below its target threshold and requires data. TRDY is automatically cleared when the data level in the TxFIFO exceeds the threshold set by TXTL bits. At reset, TRDY is set to 1. 0 The transmitter does not require data 1 The transmitter requires data (interrupt posted)
12 RTSD	RTS Delta. Indicates whether the $\overline{\text{RTS}}$ pin changed state. It (RTSD) generates a maskable interrupt. When in STOP mode, RTS assertion sets RTSD and can be used to wake the processor. The current state of the $\overline{\text{RTS}}$ pin is available on the RTSS bit. Clear RTSD by writing 1 to it. Writing 0 to RTSD has no effect. At reset, RTSD is set to 0. 0 $\overline{\text{RTS}}$ pin did not change state since last cleared 1 $\overline{\text{RTS}}$ pin changed state (write 1 to clear)
11 ESCF	Escape Sequence Interrupt Flag. Indicates if an escape sequence was detected. ESCF is asserted when the ESCEN bit is set and an escape sequence is detected in the RxFIFO. Clear ESCF by writing 1 to it. Writing 0 to ESCF has no effect. 0 No escape sequence detected 1 Escape sequence detected (write 1 to clear).
10 FRAMERR	Frame Error Interrupt Flag. Indicates that a frame error is detected. The $\overline{\textit{interrupt\_uart}}$ interrupt will be generated if a frame error is detected and the interrupt is enabled. Clear FRAMERR by writing 1 to it. Writing 0 to FRAMERR has no effect. 0 No frame error detected 1 Frame error detected (write 1 to clear)
9 RRDY	Receiver Ready Interrupt / DMA Flag. Indicates that the RxFIFO data level is above the threshold set by the RXFL bits. (See the RXFL bits description in Table 59-12 for setting the interrupt threshold.) When asserted, RRDY generates a maskable interrupt or DMA request. RRDY is automatically cleared when data level in the RxFIFO goes below the set threshold level. At reset, RRDY is set to 0. 0 No character ready 1 Character(s) ready (interrupt posted)
8 AGTIM	Ageing Timer Interrupt Flag. Indicates that data in the RxFIFO has been idle for a time of 8 character lengths (where a character length consists of 7 or 8 bits, depending on the setting of the WS bit in UCR2, with the bit time corresponding to the baud rate setting) and FIFO data level is less than RxFIFO threshold level (RXTL in the UFCR). Clear by writing a 1 to it. 0 AGTIM is not active 1 AGTIM is active (write 1 to clear)

Table 59-13. UART Status Register 1 Field Descriptions (continued)

Name	Description
7 DTRD	DTR Delta. Indicates whether $\overline{\text{DTR}}$ (in DCE mode) or $\overline{\text{DSR}}$ (in DTE mode) pins changed state. DTRD generates a maskable interrupt if DTRDEN (UCR3[3]) is set. Clear DTRD by writing 1 to it. Writing 0 to DTRD has no effect. 0 $\overline{\text{DTR}}$ (DCE) or $\overline{\text{DSR}}$ (DTE) pin did not change state since last cleared 1 $\overline{\text{DTR}}$ (DCE) or $\overline{\text{DSR}}$ (DTE) pin changed state (write 1 to clear)
6 RXDS	Receiver IDLE Interrupt Flag. Indicates that the receiver state machine is in an IDLE state, the next state is IDLE, and the receive pin is high. RXDS is automatically cleared when a character is received. RXDS is active only when the receiver is enabled. 0 Receive in progress 1 Receiver is IDLE
5 AIRINT	Asynchronous IR WAKE Interrupt Flag. Indicates that the IR WAKE pulse was detected on the RXD pin. Clear AIRINT by writing 1 to it. Writing 0 to AIRINT has no effect. Caution: AIRINT Interrupt flag cannot be used in loopback mode (UTS[12] = 1'b1) as RXD pin will be ignored (see Table 59-21). 0 No pulse was detected on the RXD IrDA pin 1 A pulse was detected on the RXD IrDA pin
4 AWAKE	Asynchronous WAKE Interrupt Flag. Indicates that a falling edge was detected on the RXD pin. Clear AWAKE by writing 1 to it. Writing 0 to AWAKE has no effect. Caution: AWAKE Interrupt flag cannot be used in loopback mode (UTS[12] = 1'b1) as RXD pin will be ignored (see Table 59-21). 0 No falling edge was detected on the RXD Serial pin 1 A falling edge was detected on the RXD Serial pin
3–0	Reserved

59.3.3.9 UART Status Register 2 (USR2)

See Figure 59-11 for illustration of valid bits in the UART Status Register 2 and Table 59-14 for description of the bit fields.

0x0098 (USR2) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADET	TXFE	DTRF	IDLE	ACST	RIDELT	RIIN	IRINT	WAKE	DCDDELT	DCDIN	RTSF	TXDC	BRCD	ORE	RDR
W	w1c		w1c	w1c	w1c	w1c		w1c	w1c	w1c		w1c		w1c	w1c	
RESET	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0

Figure 59-11. UART Status Register 2

Table 59-14. UART Status Register 2 Field Descriptions

Name	Description
31–16	Reserved
15 ADET	Automatic Baud Rate Detect Complete. Indicates that an “A” or “a” was received and that the receiver detected and verified the incoming baud rate. Clear ADET by writing 1 to it. Writing 0 to ADET has no effect. 0 ASCII “A” or “a” was not received 1 ASCII “A” or “a” was received (write 1 to clear)
14 TXFE	Transmit Buffer FIFO Empty. Indicates that the transmit buffer (TxFIFO) is empty. TXFE is cleared automatically when data is written to the TxFIFO. Even though TXFE is high, the transmission might still be in progress. 0 The transmit buffer (TxFIFO) is not empty 1 The transmit buffer (TxFIFO) is empty
13 DTRF	DTR edge triggered interrupt flag. This bit is asserted, when the programmed edge is detected on the $\overline{\text{DTR}}$ pin (DCE mode). or on $\overline{\text{DSR}}$ (DTE mode). This flag can cause an interrupt if DTREN (UCR3[13]) is enabled. 0 Programmed edge not detected on DTR/DSR 1 Programmed edge detected on DTR/DSR (write 1 to clear)
12 IDLE	Idle Condition. Indicates that an idle condition has existed for more than a programmed amount frame (refer to Section 59.4.5.1, Idle Line Detect”). An interrupt can be generated by this IDLE bit if IDEN (UCR1[12]) is enabled. IDLE is cleared by writing 1 to it. Writing 0 to IDLE has no effect. 0 No idle condition detected 1 Idle condition detected (write 1 to clear)
11 ACST	Autobaud Counter Stopped. In autobaud detection (ADBR=1), indicates the counter which determines the baud rate was running and is now stopped. This means either START bit is finished (if ADNIMP=1), or Bit 0 is finished (if ADNIMP=0). Refer to Section , New Autobaud Counter Stopped bit and Interrupt,” for more details. An interrupt can be flagged on $\overline{\textit{interrupt\_uart}}$ if ACIEN=1. 0 Measurement of bit length not finished (in autobaud) 1 Measurement of bit length finished (in autobaud). (write 1 to clear)
10 RIDELT	Ring Indicator Delta. This bit is used in DTE mode to indicate that the Ring Indicator input ($\overline{\text{RI}}$) has changed state. This flag can generate an interrupt if RI (UCR3[8]) is enabled. RIDELT is cleared by writing 1 to it. Writing 0 to RIDELT has no effect. 0 Ring Indicator input has not changed state 1 Ring Indicator input has changed state (write 1 to clear)
9 RIIN	Ring Indicator Input. This bit is used in DTE mode to reflect the status if the Ring Indicator input ($\overline{\text{RI}}$). The Ring Indicator input is used to indicate that a ring has occurred. In DCE mode this bit is always zero. 0 Ring Detected 1 No Ring Detected
8 IRINT	Serial Infrared Interrupt Flag. When an edge is detected on the RXD pin during SIR Mode, this flag will be asserted. This flag can cause an interrupt which can be masked using the control bit ENIRI: UCR4 [8]. 0 no edge detected 1 valid edge detected (write 1 to clear)
7 WAKE	Wake. Indicates the start bit is detected. WAKE can generate an interrupt that can be masked using the WKEN bit. Clear WAKE by writing 1 to it. Writing 0 to WAKE has no effect. 0 start bit not detected 1 start bit detected (write 1 to clear)
6 DCDDELT	Data Carrier Detect Delta. This bit is used in DTE mode to indicate that the Data Carrier Detect input ($\overline{\text{DCD}}$) has changed state. This flag can cause an interrupt if DCD (UCR3[9]) is enabled. When in STOP mode, this bit can be used to wake the processor. In DCE mode this bit is always zero. 0 Data Carrier Detect input has not changed state 1 Data Carrier Detect input has changed state (write 1 to clear)

Table 59-14. UART Status Register 2 Field Descriptions (continued)

Name	Description
5 DCDIN	Data Carrier Detect Input. This bit is used in DTE mode reflect the status of the Data Carrier Detect input ($\overline{DCD}$). The Data Carrier Detect input is used to indicate that a carrier signal has been detected. In DCE mode this bit is always zero. 0 Carrier signal Detected 1 No Carrier signal Detected
4 RTSF	RTS Edge Triggered Interrupt Flag. Indicates if a programmed edge is detected on the $\overline{RTS}$ pin. The RTEC bits select the edge that generates an interrupt (see Table 59-23). RTSF can generate an interrupt that can be masked using the RTSEN bit. Clear RTSF by writing 1 to it. Writing 0 to RTSF has no effect. 0 Programmed edge not detected on $\overline{RTS}$ 1 Programmed edge detected on $\overline{RTS}$ (write 1 to clear)
3 TXDC	Transmitter Complete. Indicates that the transmit buffer (TxFIFO) and Shift Register is empty; therefore the transmission is complete. TXDC is cleared automatically when data is written to the TxFIFO. 0 Transmit is incomplete 1 Transmit is complete
2 BRCD	BREAK Condition Detected. Indicates that a BREAK condition was detected by the receiver. Clear BRCD by writing 1 to it. Writing 0 to BRCD has no effect. 0 No BREAK condition was detected 1 A BREAK condition was detected (write 1 to clear)
1 ORE	Overrun Error. When set to 1, ORE indicates that the receive buffer (RxFIFO) was full (32 chars inside), and a 33rd character has been fully received. This 33rd character has been discarded. Clear ORE by writing 1 to it. Writing 0 to ORE has no effect. 0 No overrun error 1 Overrun error (write 1 to clear)
0 RDR	Receive Data Ready—Indicates that at least 1 character is received and written to the RxFIFO. If the URXD register is read and there is only 1 character in the RxFIFO, RDR is automatically cleared. 0 No receive data ready 1 Receive data ready

59.3.3.10 UART Escape Character Register (UESC)

See Figure 59-12 for illustration of valid bits in the UART Escape Character Register and Table 59-15 for description of the bit fields.

0x009c (UESC) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	ESC_CHAR							
W																
RESET	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1

Figure 59-12. UART Escape Character Register

Table 59-15. UART Escape Character Register Field Descriptions

Name	Description
31–8	Reserved
7–0 ESC_CHAR	UART Escape Character. Holds the selected escape character that all received characters are compared against to detect an escape sequence.

59.3.3.11 UART Escape Timer Register (UTIM)

See Figure 59-13 for illustration of valid bits in the UART Escape Timer Register and Table 59-16 for description of the bit fields.

0x00a0 (UTIM) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	TIM											
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 59-13. UART Escape Timer Register

Table 59-16. UART Escape Timer Register Field Descriptions

Name	Description
31–12	Reserved
11–0 TIM	UART Escape Timer. Holds the maximum time interval (in ms) allowed between escape characters. The escape timer register is programmable in intervals of 2 ms. Refer to Section 59.4.5.7, Escape Sequence Detection" and Table 59-28 for more information on the UART escape sequence detection. Reset value 0x000 = 2 ms up to 0xFFF = 8.192 s.

59.3.3.12 UART BRM Incremental Register (UBIR)

See Figure 59-14 for illustration of valid bits in the UART BRM Incremental Register and Table 59-17 for description of the bit fields.

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write 0x000F value into the UBIR after finishing detecting baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle[1].

Please note software reset will reset the register to its reset value.

0x00a4 (UBIR) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INC															
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 59-14. UART BRM Incremental Register

Table 59-17. UART BRM Incremental Register Field Descriptions

Name	Description
31–16	Reserved
15–0 INC	Incremental Numerator. Holds the numerator value minus one of the BRM ratio (refer to Section 59.4.6, Binary Rate Multiplier (BRM)"). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this field using byte accesses is not recommended and is undefined.

1. Note: The write priority in the new design is not same as the original UART. In the original design, software has higher priotiry than hardware when writing this register at the same time.

59.3.3.13 UART BRM Modulator Register (UBMR)

See Figure 59-15 for illustration of valid bits in the UART BRM Modulator Register and Table 59-18 for description of the bit fields.

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write a proper value into the UBMR based on detected baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle[1].

Please note software reset will reset the register to its reset value.

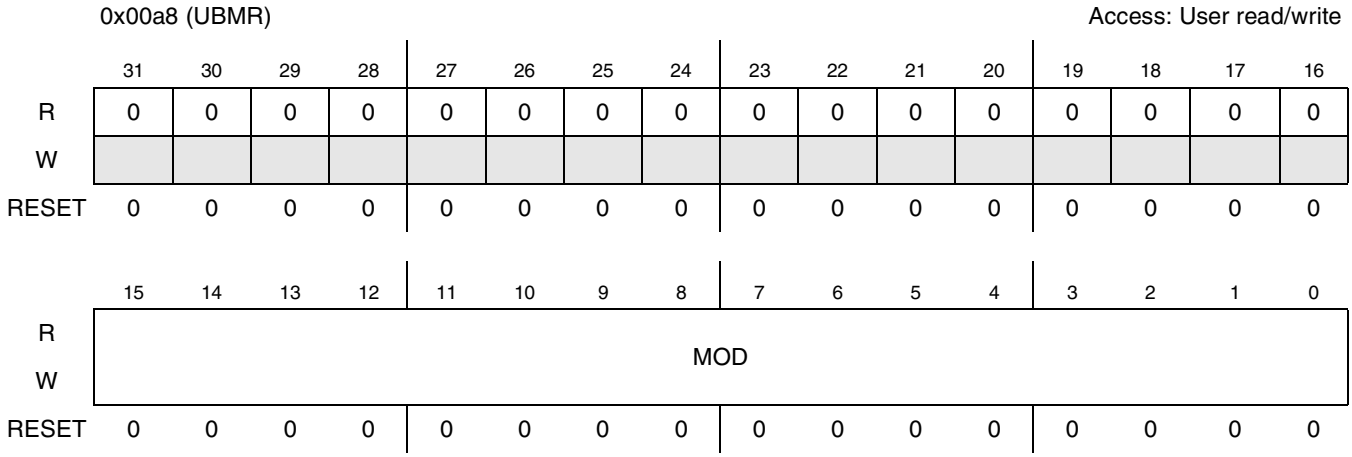


Figure 59-15. UART BRM Modulator Register

Table 59-18. UART BRM Modulator Register Field Descriptions

Name	Description
31–16	Reserved
15–0 MOD	Modulator Denominator. Holds the value of the denominator minus one of the BRM ratio (refer to Section 59.4.6, Binary Rate Multiplier (BRM)"). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this register using byte accesses is not recommended and undefined.

1. Note: The write priority in the new design is not same as the original UART. In the orginal design, software has higher priotiry than hardware when writing this register at the same time.

59.3.3.14 UART Baud Rate Count Register (UBRC)

See Figure 59-16 for illustration of valid bits in the UART Baud Rate Count Register and Table 59-19 for description of the bit fields.

0x00ac (UBRC) Access: User read

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BCNT															
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Figure 59-16. UART Baud Rate Count Register

Table 59-19. UART Baud Rate Count Register Field Descriptions

Name	Description
31–16	Reserved
15–0 BCNT	Baud Rate Count Register. This read only register is used to count the start bit of the incoming baud rate (if ADNIMP=1), or start bit + bit0 (if ADNIMP=0). When the measurement is done, the Baud Rate Count Register contains the number of UART internal clock cycles (clock after divider) present in an incoming bit. BCNT retains its value until the next Automatic Baud Rate Detection sequence has been initiated. The 16 bit Baud Rate Count register is reset to 4 and stays at hex FFFF in the case of an overflow.

59.3.3.15 UART One Millisecond Register (ONEMS)

See Figure 59-17 for illustration of valid bits in the UART One Millisecond Register and Table 59-20 for description of the bit fields.

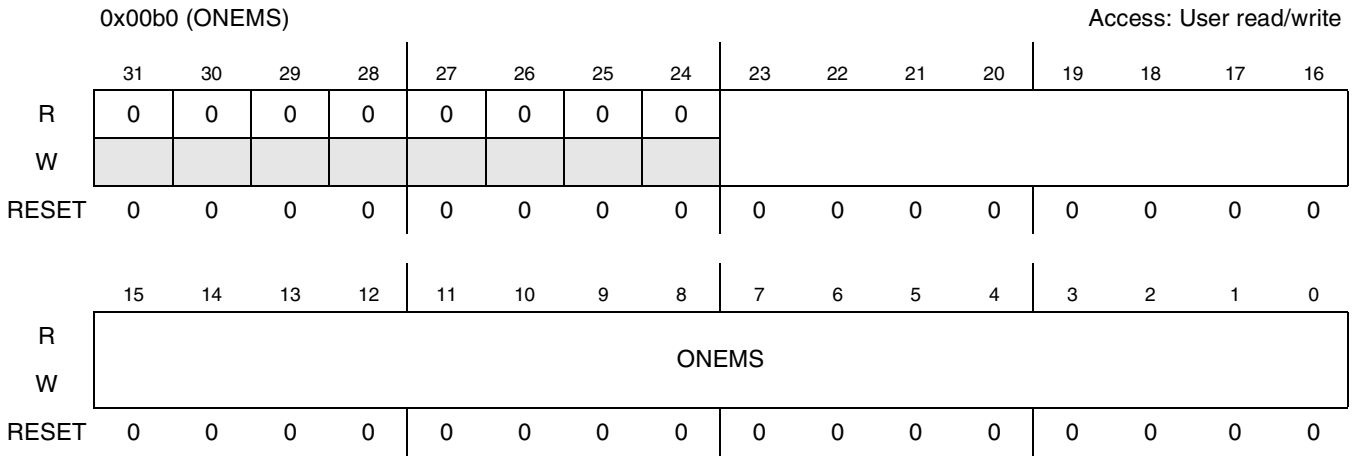


Figure 59-17. UART One Millisecond Register

Table 59-20. UART One Millisecond Register Field Descriptions

Name	Description
31–24	Reserved
23–0 ONEMS	One Millisecond Register. This 24-bit register must contain the value of the UART internal frequency (*ref_clk* in Figure 59-1) divided by 1000. The internal frequency is obtained after the UART BRM internal divider (F(*ref_clk*) = F(*module_clock*) / RFDIV). In fact this register contains the value corresponding to the number of UART BRM internal clock cycles present in one millisecond. The ONEMS (and UTIM) registers value are used in the escape character detection feature (Section 59.4.5.7, Escape Sequence Detection") to count the number of clock cycles left between two escape characters. The ONEMS register is also used in infrared special case mode (IRSC = UCR4[5] = 1'b1), refer to Section 59.4.7.3, InfraRed Special Case (IRSC) Bit."

NOTE

This register has been expanded from 16 bits to 24 bits. In previous versions, the 16-bit ONEMS can only support the maximum 65.535 MHz (0xFFFFx1000) *ref_clk*. To support 4 Mbps Bluetooth application with 66.5 MHz *module_clock*, the value 0x103C4 (66.5M/1000) should be written into this register. In this case, the 16 bits are not enough to contain the 0x103C4. So this register was expanded to 24 bits to support high frequency of the *ref_clk*.

59.3.3.16 UART Test Register (UTS)

See Figure 59-18 for illustration of valid bits in the UART Test Register and Table 59-21 for description of the bit fields.

0x00b4 (UTS) Access: User read/write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
RESET	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	FRCPERR	LOOP	DBGEN	LOOPIR	RXDBG	0	0	TXEMPTY	RXEMPTY	TXFULL	RXFULL	0	0	SOFTRST
W																
RESET	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Figure 59-18. UART Test Register

Table 59-21. UART Test Register Field Descriptions

Name	Description
31–14	Reserved
13 FRCPERR	Force Parity Error. Forces the transmitter to generate a parity error if parity is enabled. FRCPERR is provided for system debugging. 0 Generate normal parity 1 Generate inverted parity (error)
12 LOOP	Loop TX and RX for Test. Controls loopback for test purposes. When LOOP is high, the receiver input is internally connected to the transmitter and ignores the RXD pin. The transmitter is unaffected by LOOP. If RXDMUXSEL (UCR3[2]) is set to 1, the loopback is applied on serial and IrDA signals. If RXDMUXSEL is set to 0, the loopback is only applied on serial signals. 0 Normal receiver operation 1 Internally connect the transmitter output to the receiver input
11 DBGEN	
10 LOOPIR	Loop TX and RX for IR Test (LOOPIR). This bit controls loopback from transmitter to receiver in the InfraRed interface. 0 No IR loop 1 Connect IR transmitter to IR receiver
9 RXDBG	
8–7	Reserved
6 TXEMPTY	TxFIFO Empty. Indicates that the TxFIFO is empty. 0 The TxFIFO is not empty 1 The TxFIFO is empty

Table 59-21. UART Test Register Field Descriptions (continued)

Name	Description
5 RXEMPTY	RxFIFO Empty. Indicates the RxFIFO is empty. 0 The RxFIFO is not empty 1 The RxFIFO is empty
4 TXFULL	TxFIFO FULL. Indicates the TxFIFO is full. 0 The TxFIFO is not full 1 The TxFIFO is full
3 RXFULL	RxFIFO FULL. Indicates the RxFIFO is full. 0 The RxFIFO is not full 1 The RxFIFO is full
2–1	Reserved
0 SOFTRST	Software Reset. Indicates the status of the software reset ($\overline{\text{SRST}}$ bit of UCR2). 0 Software reset inactive 1 Software reset active

59.4 Functional Description

This section provides a complete functional description of the module.

59.4.1 Interrupts and DMA Requests

See Table 59-22 for the lists of all interrupt and DMA signals and associated interrupt and DMA sources of the UART. See register description section for explanation of interrupt/DMA enable and status.

Table 59-22. Interrupts and DMA

Interrupt/DMA Output	Interrupt/DMA Enable	Enable Register Location	Interrupt/DMA Flag	Flag Register Location
$\overline{\textit{interrupt\_uart}}$	RRDYEN IDEN DREN RXDSEN ATEN	UCR1 (bit 9) UCR1 (bit 12) UCR4 (bit 0) UCR3 (bit 6) UCR2 (bit 3)	RRDY IDLE RDR RXDS AGTIM	USR1 (bit 9) USR2 (bit 12) USR2 (bit 0) USR1 (bit 6) USR1 (bit 8)
$\overline{\textit{interrupt\_uart}}$	TXMPTYEN TRDYEN TCEN	UCR1 (bit 6) UCR1 (bit 13) UCR4 (bit 3)	TXFE TRDY TXDC	USR2 (bit 14) USR1 (bit 13) USR2 (bit 3)

Table 59-22. Interrupts and DMA (continued)

Interrupt/DMA Output	Interrupt/DMA Enable	Enable Register Location	Interrupt/DMA Flag	Flag Register Location
interrupt_uart	OREN BKEN WKEN ADEN ACIEN ESCI ENIRI AIRINTEN AWAKEN FRAERREN PARERREN RTSDEN RTSEN DTREN (DCE) RI (DTE) DCD (DTE) DTRDEN	UCR4 (bit 1) UCR4 (bit 2) UCR4 (bit 7) UCR1 (bit 15) UCR3 (bit 0) UCR2 (bit 15) UCR4 (bit 8) UCR3 (bit 5) UCR3 (bit 4) UCR3 (bit 11) UCR3 (bit 12) UCR1 (bit 5) UCR2 (bit 4) UCR3 (bit 13) UCR3 (bit 8) UCR3 (bit 9) UCR3 (bit 3)	ORE BRCD WAKE ADET ACST ESCF IRINT AIRINT AWAKE FRAERR PARITYERR RTSD RTSF DTRF RIDELT DCDDELT DTRD	USR2 (bit 1) USR2 (bit 2) USR2 (bit 7) USR2 (bit 15) USR2 (bit 11) USR1 (bit 11) USR2 (bit 8) USR1 (bit 5) USR1 (bit 4) USR1 (bit 10) USR1 (bit 15) USR1 (bit 12) USR2 (bit 4) USR2 (bit 13) USR2 (bit 10) USR2 (bit 6) USR1 (bit 7)
dma_req_rx	RXDMAEN ATDMAEN IDDMAEN	UCR1 (bit 8) UCR1 (bit 2) UCR4 (bit 6)	RRDY AGTIM IDLE	USR1 (bit 9) USR1 (bit 8) USR2 (bit 12)
dma_req_tx	TXDMAEN	UCR1 (bit 3)	TRDY	USR1 (bit 13)

59.4.2 Clocks

This section describes clocks and special clocking requirements of the UART.

59.4.2.1 Clock Requirements

UART module receives two clocks, *peripheral_clock* and *module_clock*. The *peripheral_clock* is used as write clock of the TxFIFO, read clock of the RxFIFO and synchronization of the modem control input pins. It must always be running when UART is enabled. There is an exception in stop mode (see Section 59.4.2.3, Clocking in Low-Power Modes").

The *module_clock* is for all the state machines, writing RxFIFO, reading TxFIFO, etc. It must always be running when UART is sending or receiving characters.This clock is used in order to allow frequency scaling on *peripheral_clock* without changing configuration of baud rate (*module_clock* staying at a fixed frequency).

The constraints on *peripheral_clock* and *module_clock* are as follows:

- *peripheral_clock* and *module_clock* can totally be asynchronous. Off course, they can also be synchronous.
- Due to the 16× oversampling of the incoming characters, *module_clock* frequency must always be greater or equal to 16× the maximum baud rate. For example, if max baud rate is 4 Mbit/s, *module_clock* must be greater or equal to 4 M × 16 = 64 MHz.

NOTE

The restriction that *peripheral_clock* frequency must be higher or equal to 16x baud rate has been removed. There is no limitation on *peripheral_clock* frequency to baud rate.

59.4.2.2 Maximum Baud Rate

The max baud rate the UART can support is determined by the max frequency of the *module_clock* and logic synthesis results. For example, if SoC can provide the fastest *module_clock* 66.5 MHz and the UART synthesis timing is OK under this constraint, the UART can transmit and receive serial data with the max baud rate 66.5M/16 = 4.15 Mbit/s.

The UART supports serial IR interface low speed. In the low speed IrDA mode, the max baud rate is 115.2 Kbit/s. To support the 115.2 Kbit/s, *module_clock* frequency must be higher or equal to 1.8432 MHz.

59.4.2.3 Clocking in Low-Power Modes

The UART supports 2 low-power modes: DOZE and STOP.

In STOP mode (input pin *stop_req* is at '1'), the UART doesn't need any clock. In this mode the UART can wake-up the MCU with the asynchronous interrupts (refer to Section 59.4.8, Low Power Modes"). An application of this feature is when the system must be waken-up by the arrival of a frame of characters.

- If before entering in Stop mode the software has enabled RTSDEN interrupt, then when RTS will change state (put at '0' by external device started to send), the asynchronous interrupt will wake-up the system, *peripheral_clock* and *module_clock* will be provided to the UART before first start bit, so that no data will be lost.
- If RTS doesn't change state (already at '0' before entering in Stop mode), then wake-up interrupt (AWAKE) will be sent at the arrival of first Start bit (on falling edge). In this case, the UART must receive the *peripheral_clock* and *module_clock* during the first half of start bit to correctly receive this character (for example, at 115.2 Kbit/s, UART must receive *peripheral_clock* and *module_clock* at maximum 4.3 microseconds after falling edge of Start bit). If the UART receives *peripheral_clock* and *module_clock* too late, first character will be lost, and so should be dropped. Also, if autobaud detection is enabled, the first character won't be correctly received and another autobaud detection will need to be initiated.

In Doze mode, UART behavior is programmable through DOZE bit (UCR1[1]). If DOZE bit is set to '1', then UART is disabled in Doze mode, and in consequence, UART clocks can be switched-off (after being sure UART is not transmitting nor receiving). On the contrary, if DOZE bit is set to '0', UART is enabled and it must receive *peripheral_clock* and *module_clock*.

59.4.3 General UART Definitions

The following list defines the general UART terms that occur in the subsequent sections.

Bit Time — The period of time required to serially transmit or receive 1 bit of data (1 cycle of the baud rate frequency).

Start bit	The bit time of a logic 0 that indicates the beginning of a data frame. A start bit begins with a 1-to-0 transition, and is preceded by at least 1 bit time of logic 1.
Stop bit	1 bit time of logic 1 that indicates the end of a data frame.
BREAK	A frame in which all of the data bits, including the stop bit, are logic 0. This type of frame is usually sent to signal the end of a message or the beginning of a new message.
Mark	When no data is being sent, the serial port's transmit pin's voltage is 1 and is said to be in a MARK state.
Space	The serial port can also be forced to keep the transmit pin at a 0 and is said to be the SPACE or BREAK state.
Frame	A start bit followed by a specified number of data or information bits and terminated by a stop bit. The number of data or information bits depends on the format specified and must be the same for the transmitting device and the receiving device. The most common frame format is 1 start bit followed by 8 data bits (least significant bit first) and terminated by 1 stop bit. An additional stop bit and a parity bit also can be included.
Framing Error	An error condition that occurs when the stop bit of a received frame is missing, usually when the frame boundaries in the received bit stream are not synchronized with the receiver bit counter. Framing errors can go undetected if a data bit in the expected stop bit time happens to be a logic 1. A framing error is always present on the receiver side when the transmitter is sending BREAKs. However, when the UART is programmed to expect two stop bits and only the first stop bit is received, this is not a framing error by definition.
Parity Error	An error condition that occurs when the calculated parity of the received data bits in a frame does not match the parity bit received on the RXD input. Parity error is calculated only after an entire frame is received.
Idle	One in NRZ encoding format and selectable polarity in IrDA mode.
Overrun Error	An error condition that occurs when the latest character received is ignored to prevent overwriting a character already present in the UART receive buffer (RxFIFO). An overrun error indicates that the software reading the buffer (RxFIFO) is not keeping up with the actual reception of characters on the RXD input.

59.4.3.1 $\overline{\text{RTS}}$—UART Request To Send

The UART Request To Send input controls the transmitter. The modem or other terminal equipment signals the UART when it is ready to receive by setting '0' on the $\overline{\text{RTS}}$ pin. Normally, the transmitter waits until this signal is active (low) before transmitting a character, however when the Ignore RTS (IRTS) bit is set, the transmitter sends a character as soon as it is ready to transmit. An interrupt (RTSD) can be posted on any transition of this pin and can wake the MCU from STOP mode on its assertion. When $\overline{\text{RTS}}$ is set to '1' during a transmission, the UART transmitter finishes transmitting the current character and shuts off. The contents of the TxFIFO (characters to be transmitted) remain undisturbed. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode.

59.4.3.2 $\overline{\text{RTS}}$ Edge Triggered Interrupt

The input to the $\overline{\text{RTS}}$ pin can be programmed to generate an interrupt on a selectable edge. See Table 59-23 for summary of the operation of the $\overline{\text{RTS}}$ edge triggered interrupt (RTSF).

To enable the $\overline{\text{RTS}}$ pin to generate an interrupt, set the request to send interrupt enable (RTSEN) bit (UCR2[4]) to 1. Writing 1 to the $\overline{\text{RTS}}$ edge triggered interrupt flag (RTSF) bit (USR2[4]) clears the interrupt flag. The interrupt can occur on the rising edge, falling edge, or either edge of the $\overline{\text{RTS}}$ input. The request to send edge control (RTEC) field (UCR2[10:9]) programs the edge that generates the interrupt. When RTEC is set to 0x00 and RTSEN = 1, the interrupt occurs on the rising edge (default). When RTEC is set to 0x01 and RTSEN = 1, the interrupt occurs on the falling edge. When RTEC is set to 0x1X and RTSEN = 1, the interrupt occurs on either edge. This is a synchronous interrupt. The RTSF bit is cleared by writing 1 to it. Writing 0 to RTSF has no effect.

Table 59-23. $\overline{\text{RTS}}$ Edge Triggered Interrupt Truth Table

$\overline{\text{RTS}}$	RTSEN	RTEC [1]	RTEC [0]	RTSF	Interrupt Occurs On…	$\overline{\textit{interrupt\_uart}}$
X	0	X	X	0	Interrupt disabled	1
1–>0	1	0	0	0	Rising edge	1
0–>1	1	0	0	1	Rising edge	0
1–>0	1	0	1	1	Falling edge	0
0–>1	1	0	1	0	Falling edge	1
1–>0	1	1	X	1	Either edge	0
0–>1	1	1	X	1	Either edge	0

There is another $\overline{\text{RTS}}$ interrupt that is not programmable. The status bit RTSD asserts the $\overline{\textit{interrupt\_uart}}$ interrupt when the $\overline{\text{RTS}}$ delta interrupt enable = 1. This is an asynchronous interrupt. The RTSD bit is cleared by writing 1 to it. Writing 0 to the RTSD bit has no effect.

59.4.3.3 $\overline{\text{DTR}}$—Data Terminal Ready

This signal indicates the general readiness of the Data Terminal Equipment (DTE). This signal is an input in DCE mode and an output in DTE mode. If the connection between the DCE and the DTE is established once, the $\overline{\text{DTR}}$ signal must remain active throughout the whole connection time. In general the $\overline{\text{DTR}}$ and $\overline{\text{DSR}}$ signals are responsible for establishing the connection. $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ are responsible for the data transfer and the transfer direction in the case of a half-duplex configuration. The $\overline{\text{DTR}}$ signal is like a “main switch”. If the $\overline{\text{DTR}}$ signal is inactive the $\overline{\text{RTS}}$ and $\overline{\text{CTS}}$ signals have no effect. In DCE mode, an interrupt (DTRD) can be posted on any transition of this pin and can wake the MCU from STOP mode on its assertion.

59.4.3.4 $\overline{\text{DSR}}$—Data Set Ready

This signal indicates the general readiness of the DCE. This signal is an output in DCE mode and an input in DTE mode. The DCE uses this signal to inform the DTE that it is switched on, has completed all preparations and can communicate with the DTE. In DTE mode, an interrupt (DTRD) can be posted on any transition of this pin and can wake the MCU from STOP mode on its assertion.

59.4.3.5 $\overline{\text{DTR/DSR}}$ Edge Triggered Interrupt

The $\overline{\text{DTR}}$ input pin (DCE mode) or $\overline{\text{DSR}}$ input pin (DTE mode) can be configured to cause an interrupt on a selectable edge. See Table 59-24 for summary of the operation of the $\overline{\text{DTR/DSR}}$ edge triggered interrupt. To enable the interrupt, set the DTREN bit (UCR3[13]) to '1'. Write a "one" to the DTRF bit (USR2[13]) to clear the interrupt flag.

The interrupt can be configured to occur on either the rising, falling, or either edge of the $\overline{\text{DTR/DSR}}$ input. Write to the DPEC[1:0] bits (UCR3[15:14]) to program which edge will cause an interrupt. If the bits are set to 00b and DTREN = 1, the interrupt will occur on the rising edge (default). If the bits are set to 01b and DTREN = 1, the interrupt will occur on the falling edge. If the bits are set to 1Xb and DTREN = 1, the interrupt will occur on either edge.

Table 59-24. $\overline{\text{DTR/DSR}}$ Edge Triggered Interrupt Truth Table

$\overline{\text{DTR}}$/$\overline{\text{DSR}}$	DTREN	DPEC[1]	DPEC[0]	DTRF	Interrupt occurs on:	*interrupt_uart*
X	0	X	X	0	Turned Off	1
1->0	1	0	0	0	Rising Edge	1
0->1	1	0	0	1	Rising Edge	0
1->0	1	0	1	1	Falling Edge	0
0->1	1	0	1	0	Falling Edge	1
1->0	1	1	X	1	Either Edge	0
0->1	1	1	X	1	Either Edge	0

59.4.3.6 $\overline{\text{DCD}}$—Data Carrier Detect

This signal is an output in DCE mode and an input in DTE mode. If used, the DCE device uses this signal to inform the DTE it has detected the carrier signal and the connection will be set up. This signal remains active while the connection remains established. In DTE mode this input can trigger an interrupt on changing state. This is achieved by setting to '1' the interrupt enable bit (DCD, UCR3[9]). The change state is reflected in DCDDELT (USR2[6]). Also, the state of the Data Carrier Detect input is mirrored in the status register DCDIN (USR2[5]).

59.4.3.7 $\overline{\text{RI}}$—Ring Indicator

This signal is an output in DCE mode and an input in DTE mode. If used, the DCE device uses this signal to inform the DTE that a ring just occurred. In DTE mode this input can trigger an interrupt on changing state. This is achieved by setting to '1' the interrupt enable bit (RI, UCR3[8]). The change state is reflected in RIDELT (USR2[10]). Also, the state of the Ring Indicator input is mirrored in the status register RIIN (USR2[9]).

59.4.3.8 $\overline{\text{CTS}}$—Clear To Send

This output pin serves two purposes. Normally, the receiver indicates that it is ready to receive data by asserting this pin (low). When the $\overline{\text{CTS}}$ trigger level is programmed to trigger at 32 characters received and

the receiver detects the valid start bit of the 33 character, it de-asserts this pin. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode.

59.4.3.9 Programmable $\overline{\text{CTS}}$ Deassertion

The $\overline{\text{CTS}}$ output can also be programmed to deassert when the RxFIFO reaches a certain level. Setting the CTS trigger level (UCR4[15:10]) at any value less than 32 deasserts the $\overline{\text{CTS}}$ pin on detection of the valid start bit of the N + 1 character (where N is the trigger level setting). However, the receiver continues to receive characters until the RxFIFO is full.

59.4.3.10 TXD—UART Transmit

This is the transmitter serial output. When operating in normal mode, NRZ encoded data is output. When operating in infrared mode, a 3/16 bit-period pulse is output for each 0 bit transmitted, and no pulse is output for each 1 bit transmitted. For RS-232 applications, this pin must be connected to an RS-232 transmitter. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode. See Figure 59-19.

59.4.3.11 RXD—UART Receive

This is the receiver serial input. When operating in normal mode, NRZ encoded data is expected. When operating in infrared mode, a narrow pulse is expected for each 0 bit received and no pulse is expected for each 1 bit received. External circuitry must convert the IR signal to an electrical signal. RS-232 applications require an external RS-232 receiver to convert voltage levels. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode. See Figure 59-19.

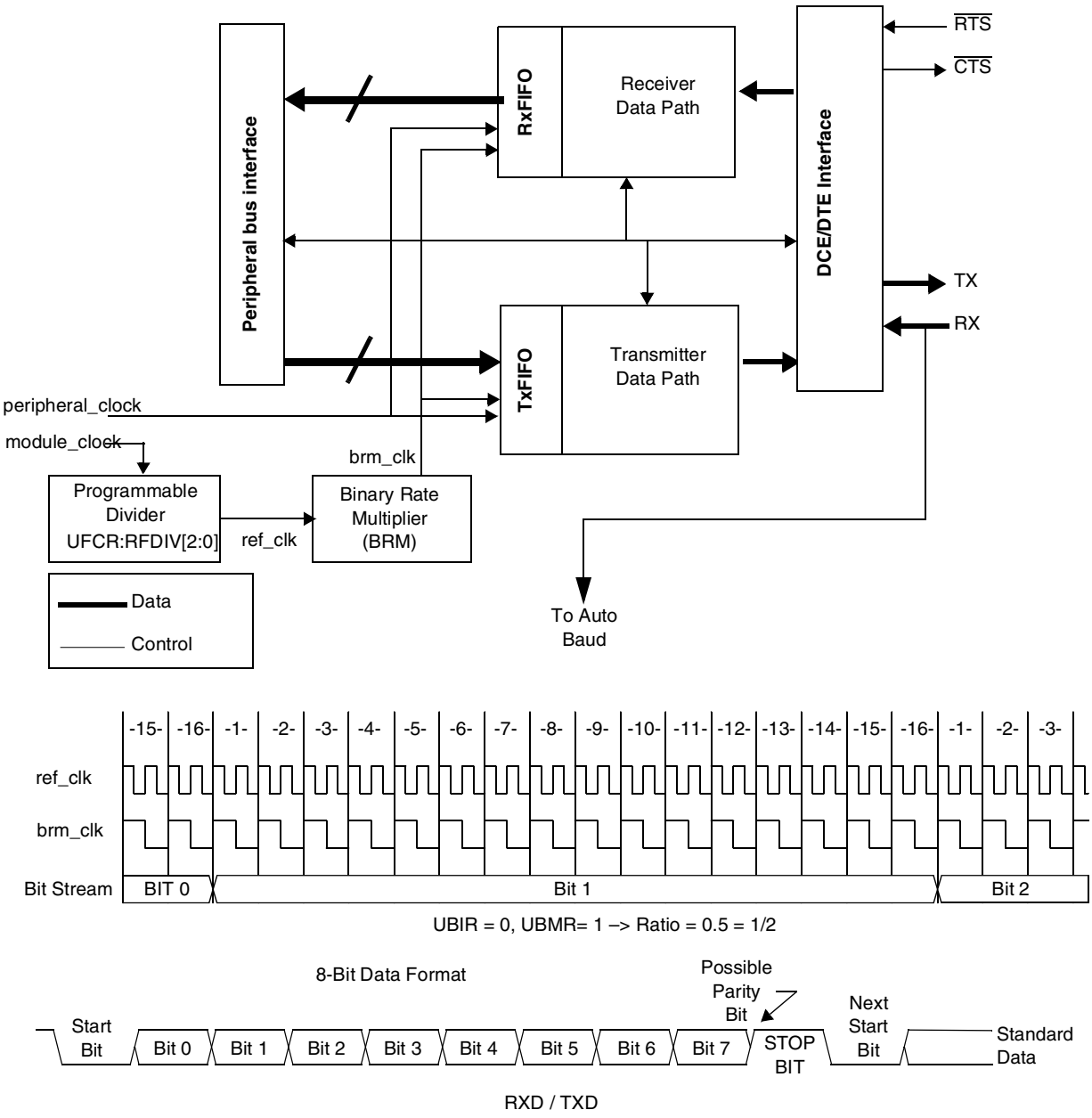


Figure 59-19. UART Simplified Block and Clock Generation Diagrams

59.4.4 Transmitter

The transmitter accepts a parallel character from the MCU and transmits it serially. The start, stop, and parity (when enabled) bits are added to the character. When the ignore RTS bit (IRTS) is set, the transmitter sends a character as soon as it is ready to transmit. $\overline{\text{RTS}}$ can be used to provide flow-control of the serial data. When $\overline{\text{RTS}}$ is set to '1', the transmitter finishes sending the character in progress (if any), stops, and waits for $\overline{\text{RTS}}$ to be set to '0' again. Generation of BREAK characters and parity errors (for debugging

purposes) is supported. The transmitter operates from the clock provided by the BRM. Normal NRZ encoded data is transmitted when the IR interface is disabled.

The transmitter FIFO (TxFIFO) contains 32 bytes. The data is written to TxFIFO by writing to the UTXD register with the byte data to the [7:0] bits. The data is written consecutively if the TxFIFO is not full. It is read (internally) consecutively if the TxFIFO is not empty. TXFULL bit (UTS[4]) can be used to control whether TXFIFO is full or not. The TxFIFO can be written regardless of the transmitter is disabled or enabled. If the UART is disabled, user can write data into the TxFIFO correctly. But in this case the write access will yield to a transfer error if the signal resp_sel is tied to 0.

59.4.4.1 Transmitter FIFO Empty Interrupt Suppression

The transmitter FIFO empty interrupt suppression logic suppresses the TXFE interrupt between writes to the TxFIFO. When TxFIFO is empty, the software can either send one or several characters. If the software sends one character, it would write the character into the UTXD register, then that character is immediately transferred to the transmitter shift register, assuming the transmitter is already enabled. Without interrupt suppression logic, the TXFE interrupt flag would be set immediately. But, with this logic, the interrupt flag is set when the last bit of the character has been transmitted, for example, before the transmission of the parity bit (if exists) and the stop bit(s).

So, the suppression logic doesn't immediately send the TXFE interrupt flag. It allows the software to write another character to the TxFIFO before the interrupt flag is asserted.

When the transmitter shift register empties before another character is written to the TxFIFO, the interrupt flag is asserted. Writing data to the TxFIFO would release the interrupt flag. The interrupt flag is asserted on the following conditions:

- System Reset
- UART software reset
- When a single character has been written to Transmitter FIFO and then the Transmitter FIFO and the Transmitter Shift Register become empty until another character is written to the Transmitter FIFO
- The last character in the TxFIFO is transferred to the shift register, when TxFIFO contains two or more characters. See Figure 59-20.

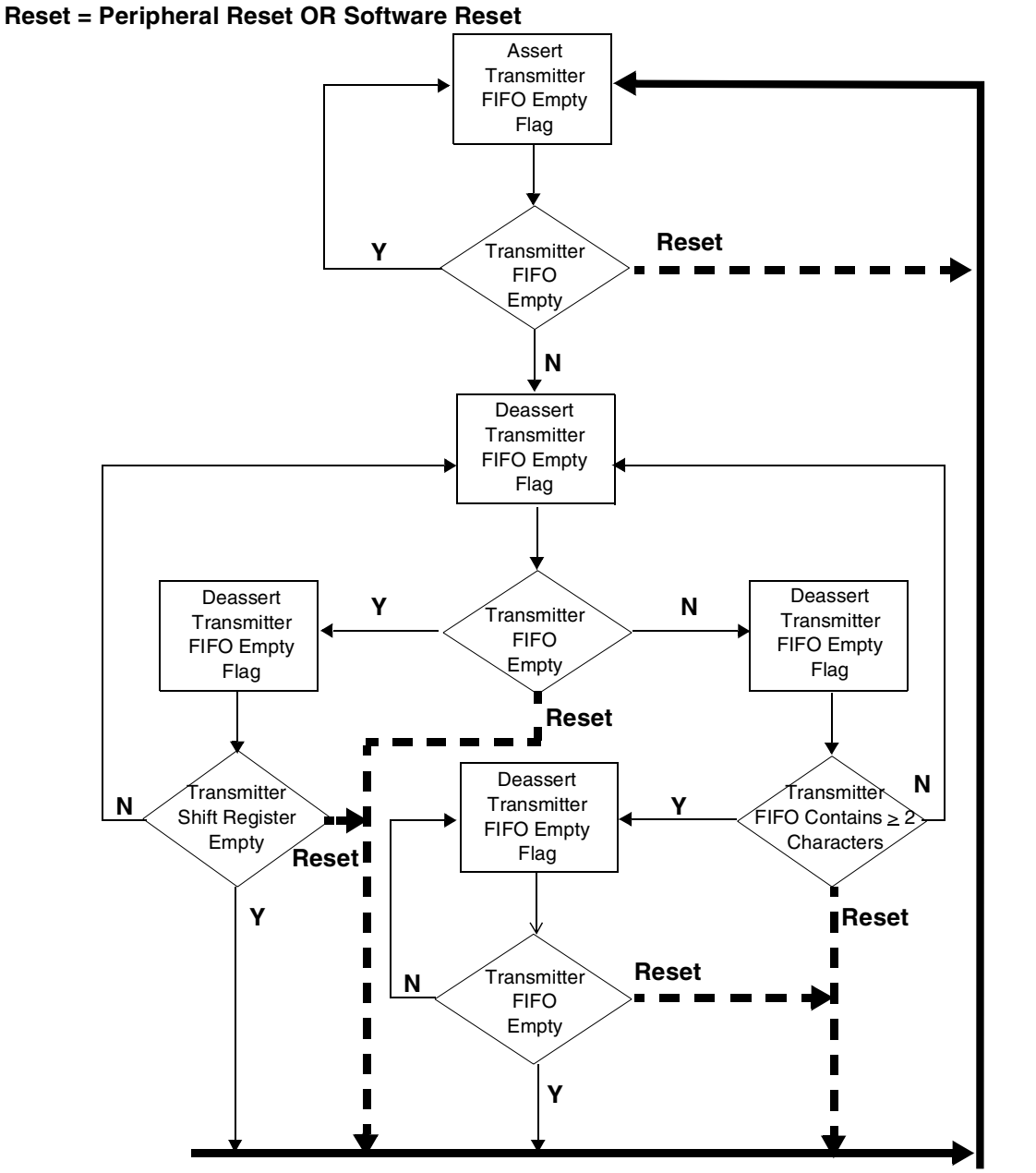


Figure 59-20. Transmitter FIFO Empty Interrupt Suppression Flow Chart

59.4.4.2 Transmitting a Break Condition

Asserting SNDBRK bit of the UCR1 Register forces the transmitter to send a break character (continuous zeros). The transmitter will finish sending the character in progress (if any) before sending break until this bit is reset. The user is responsible to ensure that this bit is high for long enough to generate a valid BREAK. The transmitter samples SNDBRK after every bit is transmitted. Following completion of the BREAK transmission, the UART will transmit two mark bits. The user can continue to fill the FIFO and any character remaining will be transmitted when the break is terminated.

59.4.5 Receiver

See Figure 59-21 for the receiver flow chart. The receiver accepts a serial data stream and converts it into parallel characters. When enabled, it searches for a start bit, qualifies it, and samples the following data bits at the bit-center. Jitter tolerance and noise immunity are provided by sampling at a 16x rate and using voting techniques to clean up the samples. Once the start bit is found, the data bits, parity bit (if enabled), and stop bits (either 1 or 2 depending on user selection) are shifted in. Parity is checked and its status reported in the URXD register when parity is enabled. Frame errors and BREAKs are also checked and reported. When a new character is ready to be read by the MCU from the RxFIFO, the receive data ready (RDR = USR2[0]) bit is asserted and an interrupt is posted (if DREN = UCR4[0] = 1). If the receiver trigger level is set to 2 (RXTL[5:0] = UFCR[5:0] = 2), and 2 chars have been received into RxFIFO, the receiver ready interrupt flag (RRDY = USR1[9]) is asserted and an interrupt is posted if the receiver ready interrupt enable bit is set (RRDYEN = UCR1[9] = 1). If the UART Receiver Register (URXD) is read once, and in consequence there is only 1 character in the RxFIFO, the interrupt generated by the RDR bit is automatically cleared. The RRDY bit is cleared when the data in the RxFIFO falls below the programmed trigger level.

Normal NRZ encoded data is expected when the IR interface is disabled. The RxFIFO contains 32 half-word entries. Characters received are written consecutively into this FIFO. If the FIFO is full and a 33rd characters is received, this character will be ignored and the USR2[ORE] bit will be set.

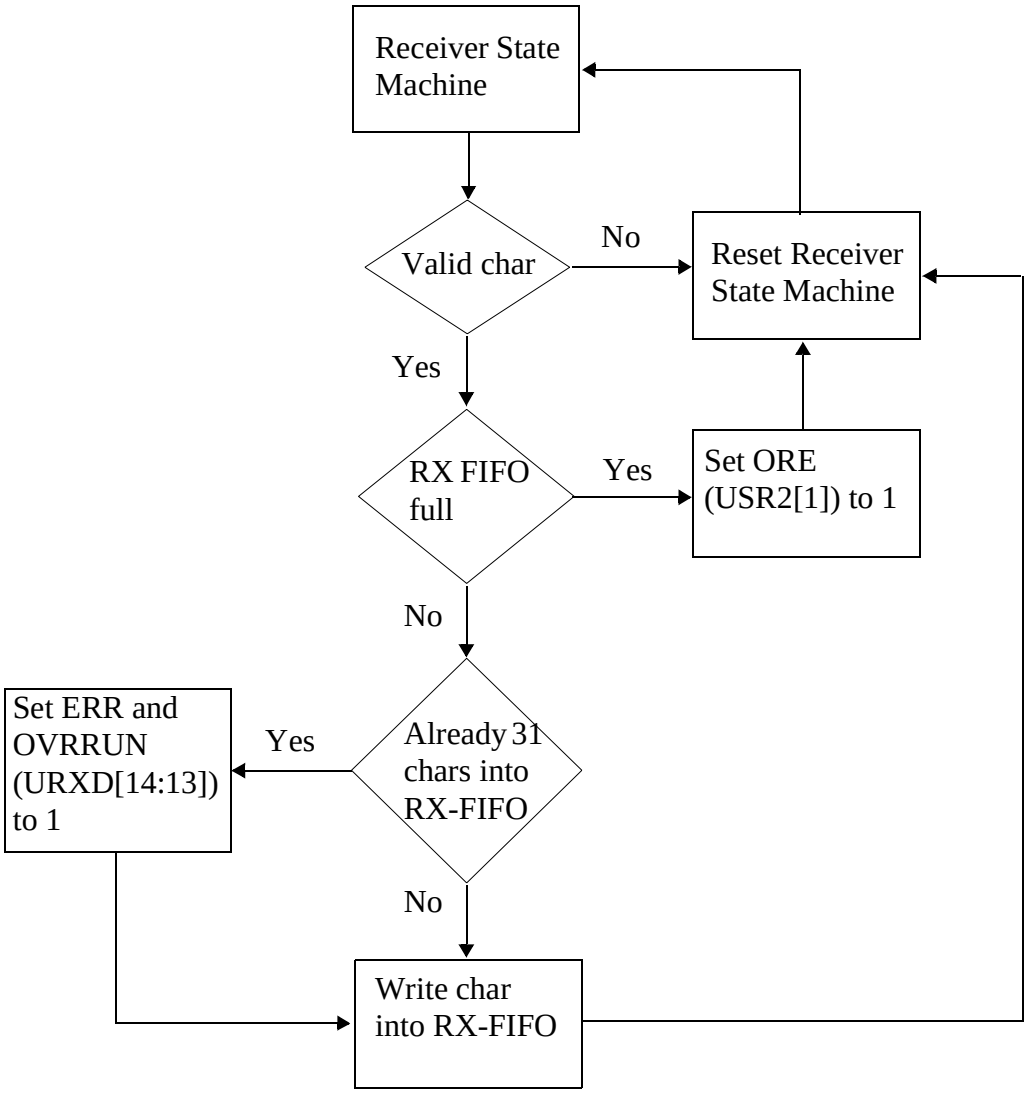


Figure 59-21. Receiver Flow Chart

59.4.5.1 Idle Line Detect

The receiver logic block includes the ability to detect an idle line. Idle lines indicate the end or the beginning of a message.

For an idle condition to occur, the following conditions must be met:

- RxFIFO must be empty
- RXD pin must be idle for more than a configured number of frames (ICD[1:0] = UCR1[11:10]).

When the idle condition detected interrupt enable (IDEN = UCR1[12]) is set and the line is idle for 4 (default), 8, 16, or 32 (maximum) frames, the detection of an idle condition flags an interrupt (see

Table 59-25). When an idle condition is detected, the IDLE (USR2[12]) bit is set. Clear the IDLE bit by writing 1 to it. Writing 0 to the IDLE bit has no effect.

Table 59-25. Detection Truth Table

IDEN	ICD [1]	ICD [0]	IDLE	*interrupt_uart*
0	X	X	0	1
1	0	0	asserted after 4 idle frames	asserted after 4 idle frames
1	0	1	asserted after 8 idle frames	asserted after 8 idle frames
1	1	0	asserted after 16 idle frames	asserted after 16 idle frames
1	1	1	asserted after 32 idle frames	asserted after 32 idle frames

Note: This table assumes that no other interrupt is set at the same time this interrupt is set for the *interrupt_uart* signal. This table shows how this interrupt affects the *interrupt_uart* signal.

During a normal message there is no idle time between frames. When all of the information bits in a frame are logic 1s, the start bit ensures that at least one logic 0 bit time occurs for each frame so that the IDLE bit is not asserted.

59.4.5.2 Ageing Character Detect

The receiver block also includes the possibility to detect when at least one character has been sitting into the RxFIFO for a time corresponding to 8 characters. This ageing character capability allows the UART to inform the MCU that there is less character into the RxFIFO than the Rx trigger and, no new character has been detected on the RXD line. The ageing capability is a timer which starts to count as soon as the RxFIFO is not empty and its trigger level is not reached (RRDY=0). This counter is reset when either a RxFIFO read is performed or another character starts to present on the RXD line. If none of those two events occurs, the bit AGTIM (USR1[8]) is set when the counter has measured a time corresponding to 8 characters. AGTIM is cleared by writing a 1 to it. AGTIM can flag an interrupt to MCU on *interrupt_uart* if ATEN (UCR2[3]) has been set.

To summarize, AGTIM is set when the following conditions are met:

- There is at least one character into RxFIFO.
- No read has occurred on RxFIFO and RXD line has stayed high, for a time corresponding to 8 characters.
- The RxFIFO trigger is not reached (RRDY=0)

59.4.5.3 Receiver Wake

The WAKE bit (USR2[7]) is set when the receiver detects a qualified Start bit. For this, two conditions must be fulfilled, firstly a falling edge on RXD line must be detected and secondly the RXD line must stays at low level for more than a half-bit duration. When the wake interrupt enable WKEN (UCR4[7]) bit is enabled, the receiver flags an interrupt (*interrupt_uart*) if the WAKE status bit is set. The WAKE bit is cleared by writing 1 to it. Writing 0 to the WAKE bit has no effect. The WAKE status bit can be asserted in either serial RS-232 mode or IR mode. The generation of the WAKE interrupt needs the clock *module_clock*.

When the asynchronous wake interrupt (AWAKE) is enabled (AWAKEN = UCR3[4] = 1), and the MCU is in STOP mode, and UART clocks have been shut-off, then a falling edge detected on the receive pin (RXD) asserts the AWAKE bit (USR1[4]) and the *interrupt_uart* interrupt to wake the MCU from STOP mode. Re-enable UART clocks and clear the AWAKE bit by writing 1 to it. Writing 0 to the AWAKE bit has no effect. When IR interface is enabled (UCR1[7]=1), the AWAKE bit is always not asserted. The generation of the asynchronous AWAKE interrupt does not need any clocks.

In IR mode, if the asynchronous IR WAKE interrupt is enabled (AIRINTEN = UCR3[5] = 1), and if MCU is in STOP mode (UART clocks are off when MCU in STOP mode), then the detection of a falling edge on the receive pin (RXD_IR), asserts the AIRINT bit (USR1[5]), and the *interrupt_uart* interrupt. This interrupt wakes the MCU from STOP mode. Software re-enables UART clocks and clear the AIRINT bit by writing 1 to it. Writing 0 to the AIRINT bit has no effect. When IR interface is disabled (UCR1[7]=0), the AIRINT bit is always not asserted. The generation of the asynchronous AIRINT interrupt does not need any clocks.

Recommended procedure for programming the asynchronous interrupts is to first clear them by writing 1 to the appropriate bit in the UART Status Register 1 (USR1). Poll or enable the interrupt for the Receiver IDLE Interrupt Flag (RXDS) in the USR1. When asserted, the RXDS bit indicates to the software that the receiver state machine is in the idle state, the next state is idle, and the RXD pin is idle (high). After following this procedure, enable the asynchronous interrupt and enter STOP mode.

59.4.5.4 Receiving a BREAK Condition

A BREAK condition is received when the receiver detects all 0s (including a 0 during the bit time of the stop bit) in a frame. The BREAK condition asserts the BRCD bit (USR2[2]) and writes only the first BREAK character to the RxFIFO. Clear the BRCD bit by writing 1 to it. Writing 0 to the BRCD bit has no effect.

Asserting BRCD would generate an interrupt on *interrupt_uart*. The interrupt generation can be masked using the control bit BKEN (UCR4[2]). Receiving a break condition will also effect the following bits in the receiver register URXD:

- URXD(11) = BRK. While high this bit indicates that the current char was detected as a break.
- URXD(12) = FRMERR. The frame error bit will always be set when BRK is set.
- URXD(10) = PRERR. If odd parity was selected the parity error bit will also be set when BRK is set.
- URXD(14) = ERR. The error detect bit indicates that the character present in the rx data field has an error status. This can be asserted by a break.

59.4.5.5 Vote Logic

The vote logic block provides jitter tolerance and noise immunity by sampling with respect to a 16x clock (*brm_clk*) and using voting techniques to clean up the samples. The voting is implemented by sampling the incoming signal constantly on the rising edge of the *brm_clk*. See Figure 59-22. The receiver is

provided with the majority vote value, which is 2 out of the 3 samples. For examples of the majority vote results of the vote logic, see Table 59-26.

Table 59-26. Majority Vote Results

Samples	Vote
000	0
101	1
001	0
111	1

The vote logic captures a sample on every rising edge of *brm_clk*, however the receiver uses 16x oversampling to take its value in the middle of the sample character.

The receiver starts to count when the Start bit is set however it does not capture the contents of the RxFIFO at the time the Start bit is set. The start bit is validated when 0s are received for 7 consecutive 1/16 of bit times following the 1-to-0 transition. Once the counter reaches 0xF, it starts counting on the next bit and captures it in the middle of the sampling frame (see Table 59-26). All data bits are captured in the same manner. Once the stop bit is detected, the receiver shift register (SIPO_OUT) data is parallel shifted to the RxFIFO.

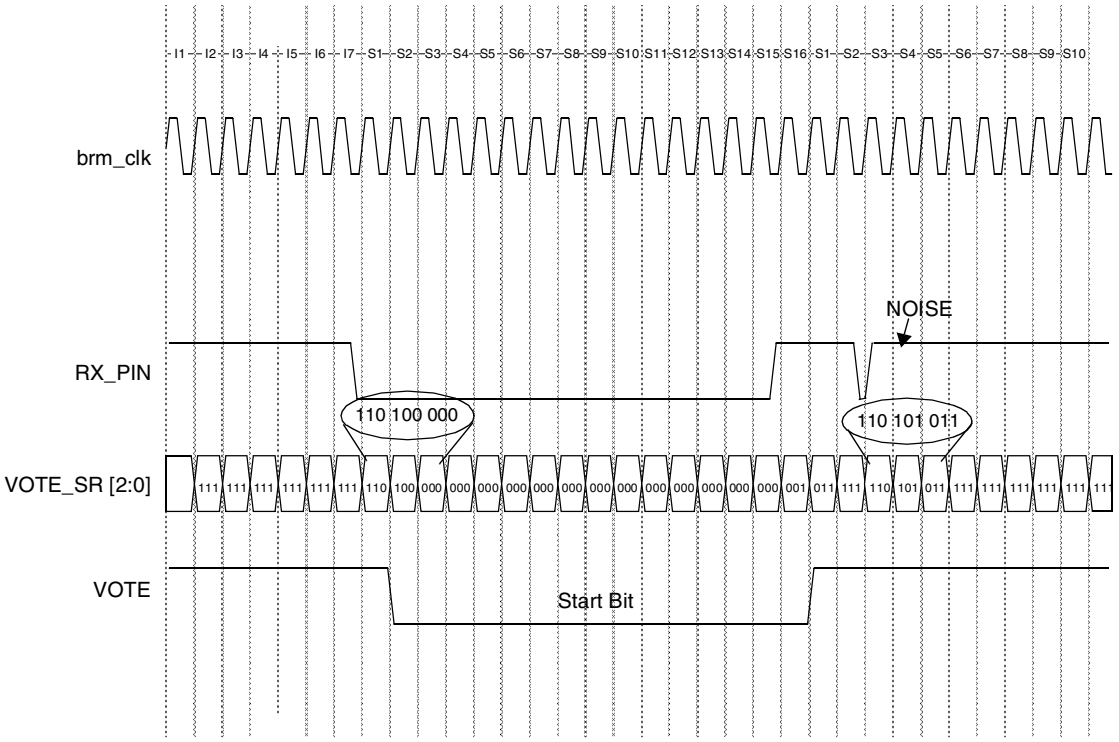


Figure 59-22. Majority Vote Results

A new feature has been recently implemented, it allows to re-synchronize the counter on each edge of RXD line. This is automatic and allows to improve the immunity of UART against signal distortion.

There is a special case when the *brm_clk* frequency is too low and is unable to capture a 0 pulse in IrDA. In this case, the software must set the IRSC (UCR4[5]) bit so that the reference clock (after internal divider) is used for the voting logic. The pulse is validated by counting the length of the pulse.

Refer to Section 59.4.7, Infrared Interface" for more details.

59.4.5.6 Baud Rate Automatic Detection Logic

When the baud rate automatic detection logic is enabled, the UART locks onto the incoming baud rate. To enable this feature, set the automatic detection of baud rate bit (ADBR = UCR1[14] = 1) and write 1 to the ADET bit (USR2[15]) to clear it. When ADET=0 and ADBR =1, the detection starts. Then, once the beginning of start bit (transition from 1-to-0 of RXD) has been detected, UART start a counter (UBRC) working at reference frequency. Once the end of start bit is detected (transition from 0-to-1 of RXD), the value of UBRC - 1 is directly copied into UBMR register. UBIR register is filled with 0x000F.

So, at the end of start bit, registers gets following values:

```
UBRC = number of reference clock periods (after divider) during Start bit.

UBIR = 0x000F

UBMR = UBRC - 1
```

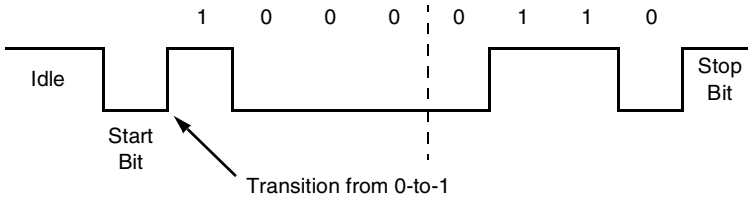
The updated values of the 3 registers can be read.

See Table 59-27 for list of parameters for baud rate detection and Figure 59-23 for baud rate detection protocol diagram.

If any of the UART BRM registers are simultaneously written by the baud rate automatic detection logic and by the peripheral data bus, the peripheral data bus would have lower priority.

Table 59-27. Baud Rate Automatic Detection

ADBR	ADET	Baud Rate Detection	$\overline{\textit{interrupt\_uart}}$
0	X	Manual Configuration	1
1	0	Auto Detection Started	1
1	1	Auto Detection Complete	0
Note: This table assumes that no other interrupt is set at the same time this interrupt is set for the $\overline{\textit{interrupt\_uart}}$ signal.			



Note: LSB transmitted first.

Figure 59-23. Baud Rate Detection Protocol Diagram

59.4.5.6.1 Baud Rate Automatic Detection Protocol

The receiver must receive an ASCII character “A” or “a” to verify proper detection of the incoming baud rate. When an ASCII character “A” (0x41) or “a” (0x61) is received and no error occurs, the Automatic Detect baud rate bit is set (ADET=1) and if the interrupt is enabled (ADEN=UCR1[15]=1), an interrupt $\overline{interrupt\_uart}$ is generated.

When an ASCII character “A” or “a” is not received (because of a bit error or the reception of another character), the auto detection sequence restarts and waits for another 1-to-0 transition.

As long as ADET = 0 and ADBR = 1, the UART continues to try to lock onto the incoming baud rate. Once the ASCII character “A” or “a” is detected and the ADET bit is set, the receiver ignores the ADBR bit and continues normal operation with the calculated baud rate.

The UART interrupt is active ($\overline{interrupt\_uart}$ = 0) as long as ADET = 1 and ADBR = 1. This can be disabled by clearing the automatic baud rate detection interrupt enable bit (ADEN = 0). Before starting an automatic baud rate detection sequence, set ADET = 0 and ADBR = 1.

The RxFIFO must contain the ASCII character “A” or “a” following the automatic baud rate detection interrupt.

The 16-bit UART Baud Rate Count Register (UBRC) is reset to 4 and stays at 0xFFFF when an overflow occurs. The UBRC register counts (measures) the duration of start bit. When the start bit is detected and counted, the UART Baud Rate Count Register retains its value until the next automatic baud rate detection sequence is initiated.

The Baud Rate Count Register counts only when auto detection is enabled.

59.4.5.6.2 Baud Rate Automatic Detection Protocol Improved

This protocol has been improved to address issues that have been reported for ICs using the existing autobaud protocol, especially for 57.6 Kbit/s and 115.2 Kbit/s. However, the old protocol is still available in the current UART IP, and several modifications can be used to make this autobaud detection more reliable. Users who wish to maintain the old method must set the bit ADNIMP (UCR3[7]) to 1. If this bit is not set (default), the autobaud improvements will be used.

The autobaud improvements are grouped in two categories: the new baud rate measurement and the new ACST bit (and associated interrupt). They are described in the following subsections.

New Baud Rate Determination

In order to fight against the problems caused by the distortion and the noise on the RXD line, the duration of the baud rate measurement has been extended. Previously, as described above, this determination was based on the measurement of the START bit duration. Now, this measurement is based on the duration of START bit + bit0. Bit0 is the first bit following the START bit. In fact, the counter which is started at the falling edge of START bit is no longer stopped at next rising edge (end of START bit), but it is stopped at the next falling edge (end of bit0). As the character sent is always a “A” (41h) or a “a” (61h), this second falling edge will always be present and it will indicate the end of bit0. Once this counter is stopped, the result is divided by 2 and used by the BRM to determine the incoming baud rate.

NOTE

UBRC register contains the result of this division by two, in consequence it reflects the measurement of the duration of one bit.

New Autobaud Counter Stopped bit and Interrupt

A new bit has been added in USR2 register: ACST (USR2[11]). This bit is set immediately after the determination of the baud rate. Settings are as follows:

- If ADNIMP is not set (default), ACST is set to 1 after the end of bit0,
- If ADNIMP is set to 1, ACST is set to 1 at the end of START bit.

If ACIEN (UCR3[0]) is set to 1, ACST flags an interrupt on $\overline{interrupt\_uart}$ signal. This interrupt informs the MCU the BRM has just been set with the result of the bit length measurement. If needed, the MCU can perform a read of UBMR (or UBRC) register and determine by itself the baud rate measured. Then the MCU has the possibility to correct the BRM registers with the nearest standardized baud rate.

NOTE

- ACST is set only if ADBR is set to 1, for example, the UART is autobauding.
- Clear the ACST bit by writing 1 to it. Writing 0 to the ACST bit has no effect.

59.4.5.7 Escape Sequence Detection

An escape sequence typically consists of 3 characters entered in rapid succession (such as +++). Because these are valid characters by themselves, the time between characters determines if it is a valid escape sequence. Too much time between two of the “+” characters is interpreted as two “+” characters, and not part of an escape sequence.

The software chooses the escape character and writes its value to the UART Escape Character Register (UESC). The software must also enable escape detection feature by setting ESCEN (UCR2[11]) to 1. The hardware compares this value to incoming characters in the RxFIFO. When an escape character is detected, the internal escape timer starts to count. The software specifies a time-out value for the maximum

allowable time between 2 successive escape characters (see Table 59-28). The escape timer is programmable in intervals of 2 ms to a maximum interval of 8.192 seconds.

Table 59-28. Escape Timer Scaling

UTIM Register	Maximum Time Between Specified Escape Characters
0x000	2 ms
0x001	4 ms
0x002	6 ms
0x003	8 ms
0x004	10 ms
…	…
0F8	498 ms
0F9	500 ms
…	…
9C3	5 s
…	…
FFD	8.188 s
FFE	8.190 s
FFF	8.192 s
Note: To calculate the time interval: (UTIM_Value + 1) × 0.002 = Time_Interval Example: (09C3 + 1) × 0.002 = 5 s.	

The escape sequence detection feature is available for all the reference frequencies. Before using the escape sequence detection, the user must fill the ONEMS register. This 24-bit register must contain the value of the UART internal frequency divided by 1000. The internal frequency is obtained after the UART internal divider which is applied on *module_clock* clock.

See the following two examples for an illustration.

Example 59-1. Example I

If the input clock *module_clock* frequency is 66.5 MHz, and if the input clock *module_clock* is divided by 2 with the internal divider UFCR[9:7] = 3’b100, the following equation is used to calculate the frequency for the ONEMS register.

Calculation of Frequency for ONEMS Register ***Eqn. 59-1***

$$ONEMS = \frac{66.5 \times 10^6}{2 \times 1000} = 33250 = 81E2h$$

Example 59-2. Example II

If the input clock *module_clock* frequency is 66.5 MHz, and if the input clock *module_clock* is divided by 1 with the internal divider: UFCR[9:7] = 3'b101, the following equation is used to calculate the frequency for the ONEMS register.

Calculation of Frequency for ONEMS Register ***Eqn. 59-2***

$$ONEMS = \frac{66.5 \times 10^6}{1000} = 66500 = 103C4h$$

The escape sequence detection interrupt is asserted when the escape sequence interrupt enable (ESCI) bit is set and an escape sequence is detected (ESCF set). Clear the ESCF bit by writing 1 to it. Writing 0 to the ESCF bit has no effect.

59.4.6 Binary Rate Multiplier (BRM)

The BRM sub-module receives *ref_clk* (*module_clock* clock after divider). From this clock, and with integer and non-integer division, BRM generates a 16x baud rate clock whose frequency is 16 times of baud rate. The uart transmitter will shift data out based on this 16x baud rate clock. The uart receiver will sample the serial data line based on this 16x baud rate clock. The input and output frequency ratio is programmed in the UART BRM Incremental Register (UBIR) and UART BRM MOD Register (UBMR). The output frequency is divided by the input frequency to produce this ratio. For integer division, set the UBIR = 0x000F and write the divisor to the UBMR register. All values written to these registers must be one less than the actual value to eliminate division by 0 (undefined), and to increase the maximum range of the registers.

Updating the BRM registers requires writing to both registers. The UBIR register must be written before writing to the UBMR register. If only one register is written to by the software, the BRM continues to use the previous values.

The following examples show how to determine what values are to be programmed into UBIR and UBMR for a given reference frequency and desired baud rate. The following equation can be used to help determine these values:

Frequency and Baud Rate for UBIR and UBMR ***Eqn. 59-3***

$$BaudRate = \frac{RefFreq}{\left(16 \times \frac{UBMR + 1}{UBIR + 1}\right)}$$

Where:

- Reference Frequency (Hz): UART Reference Frequency (*module_clock* after RFDIV divider)
- Baud Rate (bit/s): Desired baud rate.

Example 59-3. Integer Division ÷ 21

```
Reference Frequency = 19.44 MHz
UBIR = 0x000F
UBMR = 0x0014
Baud Rate = 925.7 kbit/s
```

NOTE

Observe that each value written to the registers is one less than the actual value.

Example 59-4. Non-Integer Division

```
Reference Frequency = 16 MHz
Desired Baud Rate = 920 Kbits/s
```

$$\frac{UBMR+1}{UBIR+1} = \frac{RefFreq}{16 \times BaudRate} = \frac{16 \times 10^6}{16 \times 920 \times 10^3} = 1.087$$

```
Ratio = 1.087 = 1087 / 1000
UBIR = 999 (decimal)= 0x3E7
UBMR = 1086 (decimal)= 0x43E
Non-Integer Division
Reference Frequency = 25 MHz
Desired Baud Rate = 920 kbit/s
Ratio = 1.69837 = 625 / 368
UBIR = 367 (decimal)= 0x16F
UBMR = 624 (decimal)= 0x270
```

Example 59-5. Non-Integer Division

```
Reference Frequency: 30 MHz
Desired Baud Rate = 115.2 kbit/s
Ratio = 16.276043 = 65153 / 4003
UBIR = 4002 (decimal) = 0x0FA2
UBMR = 65152 (decimal) = 0xFE80
```

59.4.7 Infrared Interface

This section discusses the infrared interface.

59.4.7.1 Generalities

The Infrared interface is selected when IREN (UCR1[7]) is set to 1.

The Infrared Interface is compatible with IrDA Serial Infrared Physical Layer Specification. In this specification, a zero is represented by a positive pulse, and a one is represented by no pulse (line remains low).

In the UART, the following is expected:

- In Tx
  - For each zero transmitted, a narrow positive pulse which is 3/16 of a bit time is generated.
  - For each one transmitted no pulse is generated (output is low). External circuitry has to be provided to drive an Infrared LED.
- In Rx
  - A narrow negative pulse is expected for each zero transmitted.
  - No pulse is expected for each one transmitted (input is high).

NOTE

The Rx part of the IR block expects to receive an inverted signal compared to IrDA specification. Circuitry external to the IC transforms the infrared signal to an electrical signal.

The IR interface has an edge triggered interrupt (IRINT). This interrupt validates a zero bit being received. This interrupt is enabled by writing a “one” to ENIRI bit.

The behavior of Infrared Interface is determined by 3 bits INVT (UCR3[1]), INVR (UCR4[9]) and IRSC (UCR4[5]).

59.4.7.2 Inverted Transmission and Reception bits (INVT and INVR)

The values of INVT and INVR depend of the IrDA transceiver connected on the TXD_IR and RXD_IR pins of the UART. If this transceiver is not inverting on both paths Tx and Rx, a zero is represented by a positive pulse and a one is represented by no pulse (line remains low). In this case, the bit INVT must be set to 0 and the bit INVR must be set to 1 (because Rx IR block expects an inverted signal).

On the contrary user must set INVT=1 and INVR=0 if both paths of the transceiver are inverting, that is, a Zero is represented as a negative pulse and a One is represented by no pulse (line remains high). The transceiver can also be inverting on only one path (Tx or Rx), in this case INVT and INVR must be together equal to 1 or to 0, depending on which path is inverted.

59.4.7.3 InfraRed Special Case (IRSC) Bit

The value to apply to IRSC bit is based on 2 parameters: the baud rate and the Minimum Pulse Duration (MPD) of the transceiver. According to IrDA Standard Specification, for SIR (Serial IR) baud rates from 2.4 Kbit/s to 115.2 Kbit/s this nominal pulse duration is equal to 3/16 of a bit duration (at the selected baud rate). But, for all the baud rates a Minimum Pulse Duration is also specified. According to IrDA Standard, a Zero is represented by a light pulse, so the IrDA transceiver can’t emit a light pulse shorter than the MPD. For SIR, the MPD is constant and equal to 1.41 us.

But user must take into account the electrical MPD associated to the transceiver on the receiver path. Typically this value is 2.0 us, but for some manufacturers MPD can go down to 1.0 us.

In order to understand the meaning of IRSC bit, one must understand how the RX path work in IrDA mode.

When UART is in IrDA mode, a Zero is not only detected by the state of the RXD_IR line, but also with the duration of the pulse. This pulse duration can be measured with two different clocks. In this case, clock is selected with the IRSC bit, as follows.

- If IRSC = 0, the clock used is the BRM clock.
- If IRSC = 1, the clock used is the UART internal clock (UART clock after the divider (RFDIV)).

In normal operation, IRSC=0. This means at any time, the user must insure the frequency of BRM_clock is high enough to measure the pulse. In the UART and for IRSC=0, the pulse must last at least 2 BRM clock cycles.

If this condition is not fulfilled, IRSC must be set to 1.

The following two examples, with the Minimum Pulse Duration equals to the MPD of the IrDA specification (in SIR), illustrate this.

Example 59-6. Calculation of BRM Clock Period (Clock Period < 1.41 µs)

The user wants to receive IrDA data at 115.2 Kbit/s. The UBIR and UBMR registers are set in order to create the BRM_clock with a frequency of 16 × baud rate = 16 × 115.2 = 1.843 MHz. But at the same time, in order to correctly detect the pulse, the user must be sure that 2 × BRM_clock period is lower than 1.41 µs.

This is checked as follows:

BRM_clock period = 1/1843000 = 542 ns

2 × BRM_clock period = 1.09 µs < 1.41 µs

Therefore, it is fine.

Example 59-7. Calculation of BRM Clock Period (Clock Period > 1.41 µs)

This time the user wants to receive at 19.2 Kbit/s. Therefore, the BRM_clock is set to 16 × 19200 = 307.2 kHz. Check whether 2 × BRM_clock period < 1.41 us as follows:

BRM_clock period =1/307200 = 3.25 µs

2 × BRM_clock period = 6.50 µs > 1.41 µs

Therefore, it does not work.

In this case, the BRM clock cannot be used to measure the pulse duration. The user must select the UART internal clock by setting IRSC = 1.

NOTE

Like for Escape character detection, when IR Special Case is enabled (IRSC=1), the UART must measure a duration. In order to do that, the user must fill the ONEMS register. Refer to Section 59.4.5.7, Escape Sequence Detection.”

59.4.7.4 IrDA interrupt

Serial infrared mode (SIR) uses an edge triggered interrupt flag IRINT (USR2[8]). When INVR =0, detection of a falling edge on the RXD pin asserts the IRINT bit. When INVR=1, detection of a rising edge on the RXD pin asserts the IRINT bit. When IRINT and ENIRI bits are both asserted, the *interrupt_uart* interrupt is asserted. Clear the IRINT bit by writing 1 to it. Writing 0 to the IRINT bit has no effect.

59.4.7.5 Conclusion about IrDA

Before using the UART in IrDA, the baud rate limit must be calculated. This baud rate limit will inform the user if IRSC bit has to be set or not.

Let's determine this limit:

As already described, if IRSC = 0, the following condition must always be fulfilled

Calculation of Baud Rate: ***Eqn. 59-4***

$$2 \times \text{BRMClockPeriod} < \text{MinPulseDuration}$$

Therefore,

$$\text{BRMClockFrequency} > \frac{2}{\text{MPD}}$$

Lnowing BRM_clock frequency = 16 × Baud Rate, we obtain:

$$\text{BaudRate} > \frac{1}{8 \times \text{MinPulseDuration}}$$

So, the user needs to set IRSC = 0 according to the following conditions:

- If Minimum Pulse Duration = 2.5 us and Baud Rate > 50 Kbit/s.
- If Minimum Pulse Duration = 2.0 us and Baud Rate > 62.5 Kbit/s.
- If Minimum Pulse Duration = 1.41 us and Baud Rate > 88.6 Kbit/s.

NOTE

For baud rates lower than the limit, IRSC must be set to 1.

59.4.7.6 Programming IrDA Interface

This section discusses the following:

- Section 59.4.7.6.1, High Speed"
- Section 59.4.7.6.2, Low Speed"

59.4.7.6.1 High Speed

The following example sequence can be used to program the IrDA interface in order to send and receive characters at 115.2 Kbit/s.

The assumptions are as follows:

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 115.2 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to MCU when 1 char is received into the Rx FIFO (RDR)

Registers values and programming orders are as follows:

- UCR1 = 0x0085
  - UCR1[7] = IREN = 1: Enable IR interface
  - UCR1[0] = UARTEN = 1: Enable UART
- UTS = 0x0000
- UFCR = 0x0981
  - TXTL[5:0] = 0x02: Default value
  - RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock is 30 MHz)
  - RXTL[5:0] = 0x01: Default value
- UBIR = 0x0202
- UBMR = 0x20BE, Baud rate = 115.2 kbit/s with internal clock = 30 MHz
- UCR2 = 0x4027
  - UCR2[14] = IRTS = 1: Ignore level of RTS input signal
  - UCR2[5] = WS = 1: Characters are 8-bit length
  - UCR2[2] = TXEN = 1: Enable Rx path
  - UCR2 [1] = RXEN = 1: Enable Tx path
  - UCR2[0] = SRST_B = 1: No software reset
- UCR3 = 0x0000
- UCR4 = 0x8201
  - CTSTL[5:0] = 0x20: Default value
  - UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
  - UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)

The UART is ready to send a character as soon as there is a write into UTXD register. An interrupt will be sent to MCU when a character is received.

59.4.7.6.2 Low Speed

This example sequence uses the same assumptions as in Section 59.4.7.6.1, High Speed, ″ but the speed is now 9.6 Kbit/s. This baud rate is below the limit (even with a minimum pulse duration of 2.5 µs); thus IRSC must be set to 1.

The assumptions are as follows:

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 9.6 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to MCU when 1 char is received into the Rx FIFO (RDR).

Registers values and programming orders are as follows:

- UCR1 = 0x0085
  - UCR1[7] = IREN = 1: Enable IR interface
  - UCR1[0] = UARTEN = 1: Enable UART
- UFCR = 0x0981
  - UFCR[15:10] = TXTL[5:0] = 0x02: Default value
  - RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock is 30 MHz)
  - UFCR[5:0] = RXTL[5:0] = 0x01: Default value
- UBIR = 0x00FF
- UBMR = 0xC354 Baud rate = 9.6 kbit/s with internal clock = 30 MHz
- UCR2 = 0x4027
  - UCR2[14] = IRTS = 1: Ignore level of RTS input signal
  - UCR2[5] = WS = 1: Characters are 8-bit length
  - UCR2[2] = TXEN = 1: Enable Rx path
  - UCR2 [1] = RXEN = 1: Enable Tx path
  - UCR2[0] = SRST_B = 1: No software reset
- UCR3 = 0x0000
  - UCR3[1] = INVT = 0: Positive pulse represents 0.
- UCR4 = 0x8221
  - UCR4[15:10] = CTSTL[5:0] = 0x20: Default value
  - UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
  - UCR4[5] = IRSC = 1: Because data rate is below the limit and thus the UART internal clock is used to measure the pulse duration.
  - UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)

The UART is now ready to send a character as soon as there is a write into UTXD register. An interrupt will be sent to MCU when a character is received.

59.4.8 Low Power Modes

Table 59-29 shows the UART functionality while in hardware controlled low-power modes. These modes are controlled by the signals *doze_req* and *stop_req*. The control/status/data registers do not change when entering or exiting low power modes.

Table 59-29. UART Low Power State Operation

	Normal State (*doze_req* = 1'b0 & *stop_req* = 1'b0)	Doze State (*doze_req* = 1'b1)		Stop State (*stop_req* = 1'b1)
		DOZE bit = 0	DOZE bit = 1	
UART-Clock	ON	ON	ON	OFF
UART Serial / IrDA	ON	ON	OFF	OFF

59.4.8.1 UART Operation in System Doze Mode

While in Doze State (when *doze_req* input pin is set to 1'b1), the UART behavior depends on the DOZE (UCR1[1]) control bit. While the DOZE bit is negated, the UART serial interface is enabled. While the system is in the Doze State, and the DOZE bit is asserted, the UART is disabled. If the Doze State is entered with the DOZE bit asserted while the UART serial interface was receiving or transmitting data, it will complete the receive/transmit of the current character and signal to the far-end transmitter/receiver to stop sending/receiving.

59.4.8.2 UART Operation in System Stop Mode

The internal baud rate clocks of the transmitter and receiver are gated off if the *stop_req* signal to UART is asserted. Even though the clocks at the input of UART module continue to run during system Stop mode the UART will not do any transmission or reception.

The following UART interrupts wake the MCU processor from STOP mode:

- RTS (RTSD)
- IrDA Asynchronous WAKE (AIRINT)
- Asynchronous WAKE (AWAKE)
- RI (RIDELT in DTE mode only)
- DCD (DCDDELT in DTE mode only)
- DTR (DTRD in DCE mode only)
- DSR (DTRD in DTE mode only)

When an asynchronous WAKE (awake) interrupt exits the MCU from STOP mode, make sure that a dummy character is sent first because the first character may not be received correctly.

59.4.8.3 Power Saving Method in UART

The RXEN (UCR2[1]), TXEN (UCR2[2]) and UARTEN (UCR1[0]) bits are set by the user and provide software control of low-power modes.

Setting the UARTEN (UCR1[0]) bit to 0 shuts off the receiver and transmitter logic and the associated clocks.

If the UART is used only in transmit mode, UARTEN and TXEN must be set to 1. If the UART is used only in receive mode, UARTEN and RXEN must be set to 1. Setting TXEN or RXEN to 0 allows to save a lot of power.

59.4.9 Reset

This section describes how to reset the module and explains special requirements related to reset.

59.4.9.1 Hardware reset

All of registers, FIFOs, state machines, and sequential elements can be reset to their initial values by hardware reset or power on reset.

59.4.9.2 Software reset

The status registers USR1 and USR2, BRM registers UBIR and UBMR, TxFIFO and RxFIFO, and transmitter and receiver state machines can be reset by software reset. Internal logic will remain the software reset signal at active for about 4 *module_clock* cycles. Programmer can follow the following software reset sequence:

1. Clear the $\overline{\text{SRST}}$ bit (UCR2[0])
2. Wait for software reset complete: poll SOFTRST bit (UTS[0]) until it is 0.
3. Re-program baud rate registers: Re-write UBIR and UBMR.

59.4.10 Transfer Error

The UART can generate a transfer error on the peripheral bus in the following cases:

- Core is writing into a read-only register.
- Core is accessing (read or write) an unused location within the assigned address space reserved to UART.
- Core is writing into UTXD register with transmit interface disabled (TXEN=0 or UARTEN=0)
- Core is reading URXD register with receive interface disabled (RXEN=0 or UARTEN=0)

59.4.11 Functional Timing

This section includes timing diagrams for functional signaling.

59.4.11.1 RS-232 Mode

When transmitting a byte, the UART first sends a Start Bit which is logic 0, followed by the data (general 8 bits, but could be 7 bits) followed by patity bit (optional) and one or two Stop Bits which is logic 1. The

sequence is repeated for each byte sent. Figure 59-24 shows a diagram of a what a byte transmission would look like.The receiver also sample data according to this format.

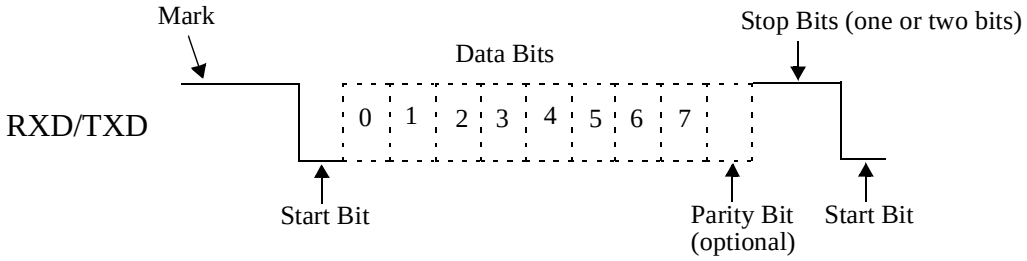


Figure 59-24. Timing Diagram of RS-232 Serial Data Line

59.4.11.2 IrDA Mode

According to IrDA specification, the low speed (115.2 Kbit/s and below) IR frame format is compatible with UART frame. Figure 59-25 shows the timing of IR data line and corresponding UART frame. In this figure, an example data 0x65 is used.

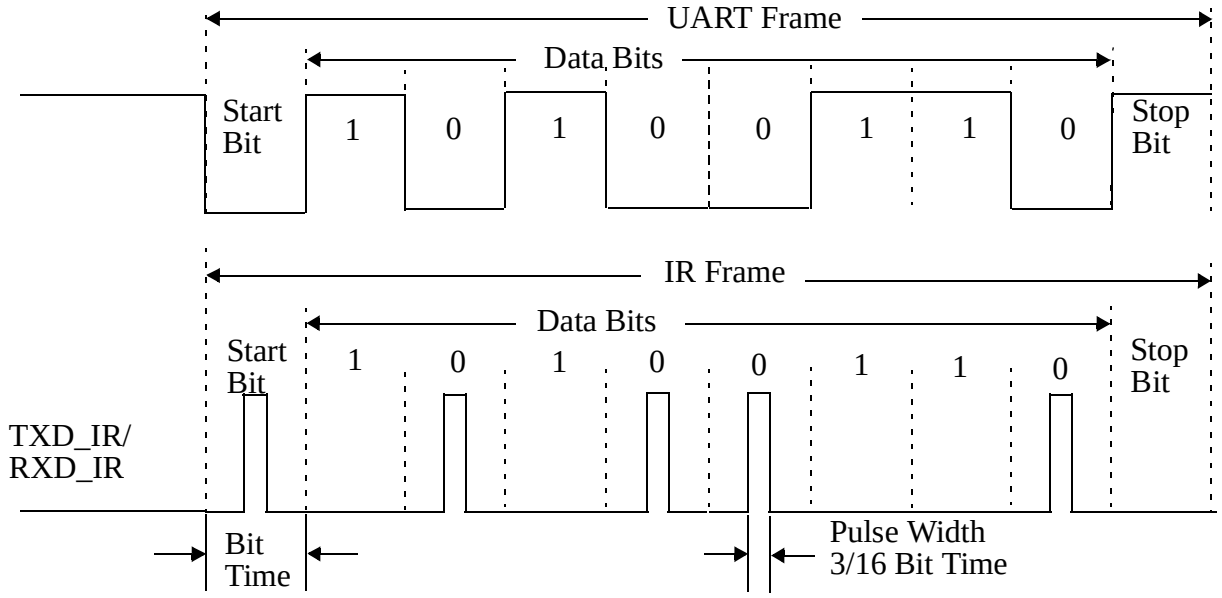


Figure 59-25. Timing diagram of Low Speed IR (≤115.2 Kbit/s) Data Line

59.5 Initialization

The following example sequence can be used to program the UART in order to send and receive characters in RS-232 mode.

The assumptions are as follows:

- Input UART clock = 100 MHz
- Baud rate = 921.6 Kbps
- Data bits = 8 bits
- Parity = Even
- Stop bits = 1 bit
- Flow control = Hardware

Main program:

1. UCR1 = 0x0001

Enable the UART.

2. UCR2 = 0x2127

Set hardware flow control, data format and enable transmitter and receiver.

3. UCR3 = 0x0704

Set UCR3[RXDMUXSEL] = 1.

4. UCR4 = 0x7C

Set CTS trigger level to 31,

5. UFCR = 0x089E

Set internal clock divider = 5 (divide input UART clock by 5). So the reference clock is 100 MHz ÷ 5 = 20 MHz.

Set TXTL = 2 and RXTL = 30.

6. UBIR = 0x08FF
7. UBMR = 0x0C34

In the above two steps, set baud rate to 921.6 Kbps based on the 20 MHz reference clock.

8. UCR1 = 0x2201

Enable the TRDY and RRDY interrupts.

Interrupt service routine for the transmitter:

- Write characters into UTXD

The TRDY interrupt is automatically de-asserted when the data level of the TxFIFO exceeds the TXTL = 2. Note that the first time, the interrupt may be de-asserted after four characters are written into the TxFIFO because of the shift register.

Interrupt service routine for the receiver:

- Read characters from URXD

The RRDY interrupt will be automatically de-asserted when the data level of the RxFIFO is below the RXTL = 30.

Chapter 60
Universal Serial Bus OTG HOST (USBOH3)

60.1 Introduction

The USBOH3 module provides high performance USB On-The-Go (OTG) functionality that conforms to the USB 2.0 specification, the OTG supplement, and the ULPI specification. The module consists of four independent USB cores, each with serial and ULPI USB ports. Three of the USB host controllers are identical.

In addition to the normal USB functionality, the module also supports direct connections to on-board USB peripherals using serial or ULPI protocol. It also has serial/ULPI bypass mode connection and support for multiple interface types of ULPI and serial transceivers.

It provides for Full-Speed Transceiverless Link (FS-TLL) operation on host port 1, port 2, and port 3. High-Speed Transceiverless Link (HS-TLL) operation is possible on all four ports and the IC-USB interface on host ports 1, 2, and 3, which permits routing of the OTG transceiver interface to host port 1. This allows this transceiver to be used to communicate with a USB peripheral connected to host port 1.

Figure 60-1 shows a block diagram of USBOH3.

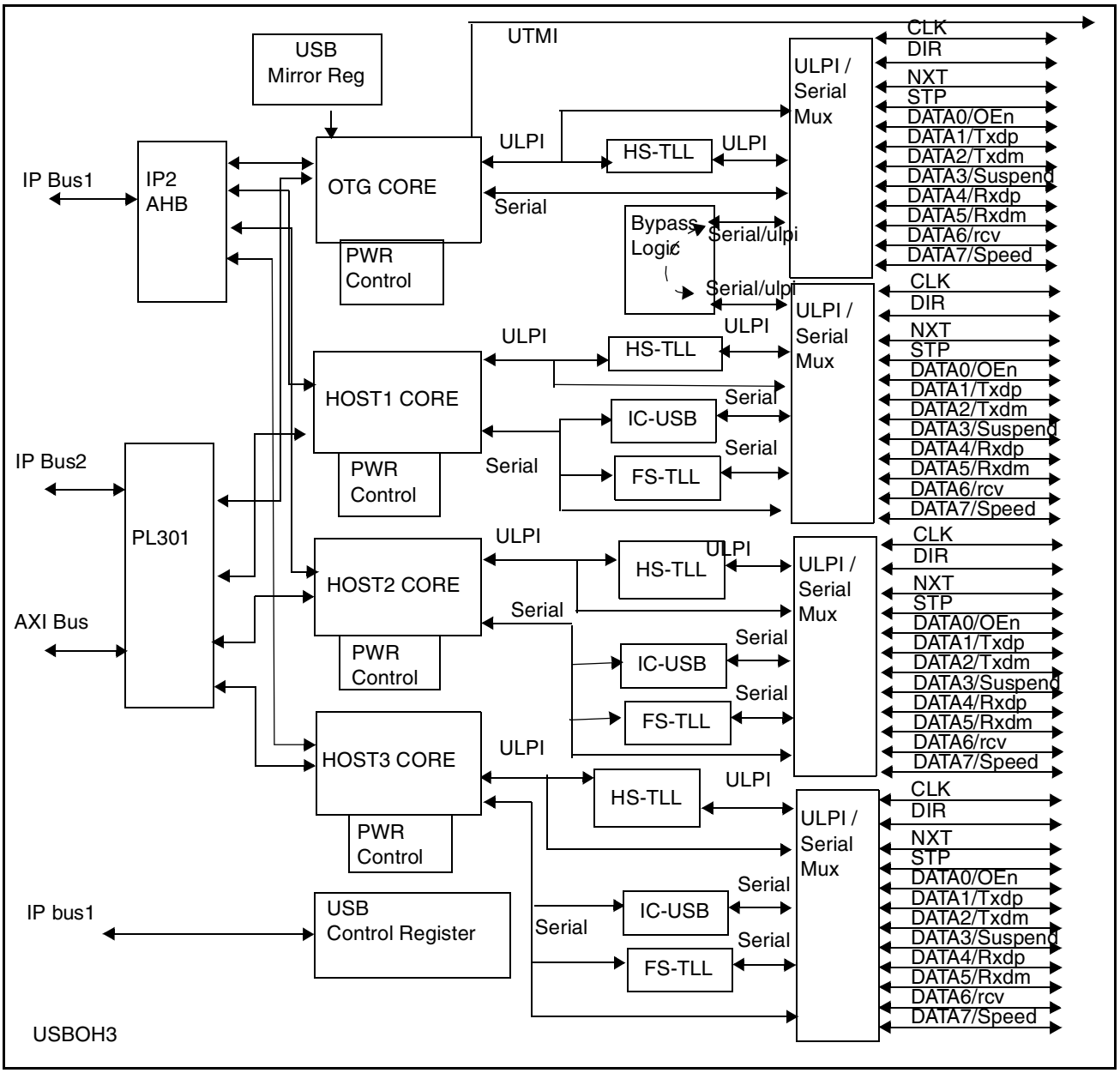


Figure 60-1. USBOH3 Block Diagram

60.1.1 Features

The USBOH3 module includes the following features:

- High-speed/full-speed/low-speed host only core (HOST 1)
  - — HS/FS ULPI compliant interface
  - — Software configurable for full-speed/low-speed interface for serial transceiver

- Full-Speed transceiverless link logic (FS-TLL) for onboard connection to a FS/LS USB peripheral
- High-speed ULPI transceiverless link logic (HS-TLL) for onboard connection to a high-speed ULPI interface USB peripheral (ULPI Link)
- Bypass mode to route host port 1 signals to OTG I/O port by Serial or ULPI mode

- High-speed/full-speed/low-speed host only core (HOST2)
  - HS/FS ULPI compliant interface
  - Software configurable for full speed/low speed interface for serial transceiver
  - Full-speed transceiverless link logic (FS-TLL) for on board connection to a FS/LS USB peripheral
  - High-speed ULPI transceiverless link logic (HS-TLL) for on board connection to a high-speed ULPI interface USB peripheral
- High-speed/full-speed/low-speed Host Only core (HOST3)
  - HS/FS ULPI compliant interface
  - Software configurable for full-speed/low-speed interface for serial transceiver
  - Full-speed transceiverless link logic (FS-TLL) for on board connection to a FS/LS USB peripheral
  - High-speed ULPI transceiverless link logic (HS-TLL) for onboard connection to a high-speed ULPI interface USB peripheral
- High-speed OTG core
  - HS/FS ULPI compliant interface
  - Software configurable for ULPI or serial transceiver interface
  - High-speed (with ULPI transceiver), full-speed, and low-speed operation in host mode
  - High-speed (with ULPI transceiver), and full-speed operation in peripheral mode
  - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
  - Up to 8 bidirectional endpoints
- Low-power mode with local and remote wake-up capability
- Serial PHY interfaces configurable for bidirectional/unidirectional and differential/single ended
- Embedded DMA controller

60.1.2 Modes of Operation

The USBOH3 module has two main modes of operation: normal mode and bypass mode. Furthermore, each USB core interface can be configured for high-speed operation (480 Mbps) and/or full-/low-speed operation (12/1.5 Mbps).

60.1.2.1 Operational Modes

The following subsections describe the configuration options in greater detail.

60.1.2.1.1 Normal Mode

In normal mode, each USB core controls its corresponding port. Each port can work in one or more modes.

- Host Ports 1, 2, and 3

  Each port supports ULPI and serial transceivers.

  — Serial interface mode

  PHY mode—for connections using transceivers.

  TLL mode—for direct onboard connections to FS USB peripherals.

  — ULPI interface mode

  ULPI is the low-pin count standard for connecting off-chip high-speed USB transceivers to a USB device. When the port is configured for ULPI mode, only a ULPI compatible transceiver can be used

  — IC-USB mode

  Host can connect with IC-USB device

  — HS/FS TLL mode

  In TLL mode, internal logic is enabled to emulate the functionality of two back-to-back connected transceivers. The HS-TLL is used in ULPI interface for high speed transfer. and the FS-TLL is used in serial interface for full/low speed transfer. These two modes are typically used for on-board USB connections to USB-capable peripherals

- OTG port

  This Port requires a transceiver and is intended for off-board USB connections.

  — Serial Interface mode

  In serial mode, a serial OTG transceiver must be connected. The port does not support dedicated signals for OTG signaling. Instead, a transceiver with built-in OTG registers must be used. Typically, the transceiver registers are accessible over an I^2C or SPI interface

  — ULPI Mode

  It this mode, a ULPI transceiver is connected to the port pins to support High-speed off-chip USB connections. The ULPI mode is activated by writing to the appropriate register

  — HS TLL mode

  In HS-TLL mode, internal logic is enabled to emulate the functionality of two back-to-back connected transceivers. The OTG port can only support HS-TLL

NOTE

Since ULPI and Serial mode do not operate at the same time they share the ULPI_DATA[7:0] signals as the serial interface signals. Table 60-1 shows the signal mapping for serial interface in four different serial modes.

Table 60-1. ULPI_DATA[7:0] Signal Mapping

	Unidirectional Differential	Unidirectional Single-End	Bidirectional Differential	Bidirectional Single-End
ulpi_data[0]	oen	oen	oen	oen

Table 60-1. ULPI_DATA[7:0] Signal Mapping

ulpi_data[1]	txdp	tx_dat	dp	dat
ulpi_data[2]	txdm	tx_se0	dm	se0
ulpi_data[3]	suspend	suspend	suspend	suspend
ulpi_data[4]	rxdp	rxdp	/	/
ulpi_data[5]	rxdm	rxdm	/	/
ulpi_data[6]	rcv	rcv	rcv	/
ulpi_data[7]	speed	speed	speed	speed

60.1.2.1.2 Bypass Mode

Bypass mode affects the operation of the OTG port and host port 1. This mode is only available when a serial/ULPI transceiver is used on the OTG port and the peripheral device on host 1 port is using a TLL connection with same type port.

Bypass mode is activated by setting the bypass bit in the USBCONTROL register. In this mode, the USB OTG port connections are internally routed to the USB HOST 1 port, such that the transceiver on the OTG port connects to a peripheral USB device attached to the HOST 1 port. The OTG core and the HOST 1 core are disconnected from their ports when bypass is active. When in serial bypass mode, the status of the DM and DP inputs for the cores is programmable in the USB Control Register (Table 60-2).

60.1.2.1.3 Low-Power Mode

Each of the three USB cores has an associated power control module that is controlled by the USB core and clocked using a 32 kHz clock. When a USB bus is idle, the transceiver can be placed in low-power mode (suspend), after which the clocks to the USB core can be stopped. The 32 kHz low-power clock must remain active as it is needed for wakeup detection.

Either the local CPU or the remote USB Host/Peripheral can initiate a wake-up sequence to resume USB communication.

60.2 Memory Map/Register Definition

Table 60-2 shows the USBOH3 module memory map.

Table 60-2. USBOH3 Module Memory Map

Address	Controller	Use	Access
Base + 0x000	OTG	ID (UOG_ID)	RO
Base + 0x004	OTG	Hardware General (UOG_HWGENERAL)	RO
Base + 0x008	OTG	Host Hardware Parameters (UOG_HWHOST)	RO
Base + 0x010	OTG	TX Buffer Hardware Parameters (UOG_HWTXBUF)	RO
Base +0x014	OTG	RX Buffer Hardware Parameters (UOG_HWRXBUF)	RO
Base +0x080	OTG	General Purpose Timer #0 Load(UOG_GPTIMER0LD)	RW

Table 60-2. USBOH3 Module Memory Map (continued)

Address	Controller	Use	Access
Base +0x084	OTG	General Purpose Timer #0 Controller(UOG_GPTIMER0CTRL)	RW
Base +0x088	OTG	General Purpose Timer #1 Load(UOG_GPTIMER0LD)	RW
Base +0x08c	OTG	General Purpose Timer #1 Controller(UOG_GPTIMER0CTRL)	RW
Base +0x100	OTG	Capability Register Length (UOG_CAPLENGTH)	RO
Base + 0x102	OTG	Host Interface Version (UOG_HCIVERSION)	RO
Base +0x104	OTG	Host Control Structural Parameters (UOG_HCSPARAMS)	RO
Base +0x108	OTG	Control Capability Parameters (UOG_HCCPARAMS)	RO
Base + 0x120	OTG	Device Interface Version (UOG_DCIVERSION)	RO
Base + 0x124	OTG	Device Controller Capability Parameters (UOG_DCCPARAMS)	RO
Base +0x140	OTG	USB Command Register (UOG_USBCMD)	RW
Base +0x144	OTG	USB Status Register (UOG_USBSTS)	RW
Base +0x148	OTG	Interrupt Enable Register (UOG_USBINTR)	RW
Base + 0x14C	OTG	USB Frame Index (UOG_FRINDEX)	RW
Base +0x154	OTG	Host Controller Frame List Base Address (UOG_PERIODICLISTBASE)	RW (32-bit only)
Base +0x158	OTG	Host Controller Next Asynch. Address (UOG_ASYNCLISTADDR)	RW (32-bit only)
Base +0x160	OTG	Host Controller Embedded TT Asynch. Buffer Status (UOG_BURSTSIZE)	RW (32-bit only)
Base +0x164	OTG	TX FIFO Fill Tuning (UOG_TXFILLTUNING)	RW (32-bit only)
Base +0x16C	OTG	IC_USB enable and voltage negotiation (UOG_IC_USB)	RW
Base + 0x170	OTG	ULPI Viewport (UOG_ULPIVIEW)	RW
Base +0x180	OTG	Config Flag (UOG_CFGFLAG)	RO
Base +0x184	OTG	Port Status & Control (UOG_PORTSC1)	RW
Base + 0x1A4	OTG	On-The-Go Status & control (UOG_OTGSC	RW
Base +0x1A8	OTG	USB Device Mode (UOG_USBMODE)	RW
Base + 0x1AC	OTG	Endpoint Setup Status (UOG_ENDPTSETUPSTAT)	RW
Base + 0x1B0	OTG	Endpoint Initialization (UOG_ENDPTPRIME)	RW
Base + 0x1B4	OTG	Endpoint De-Initialize (UOG_ENDPTFLUSH)	RW
Base + 0x1B8	OTG	Endpoint Status (UOG_ENDPTSTAT)	RO
Base + 0x1BC	OTG	Endpoint Complete (UOG_ENDPTCOMPLETE)(	RW
Base + 0x1C0	OTG	Endpoint Control0 (UOG_ENDPTCTRL0)	RW
Base + 0x1C4	OTG	Endpoint Control1 (UOG_ENDPTCTRL1)	RW
Base + 0x1C8	OTG	Endpoint Control2 (UOG_ENDPTCTRL2)	RW
Base + 0x1CC	OTG	Endpoint Control3 (UOG_ENDPTCTRL3)	RW
Base + 0x1D0	OTG	Endpoint Control4 (UOG_ENDPTCTRL4)	RW
Base + 0x1D4	OTG	Endpoint Control5 (UOG_ENDPTCTRL5)	RW
Base + 0x1D8	OTG	Endpoint Control6 (UOG_ENDPTCTRL6)	RW
Base + 0x1DC	OTG	Endpoint Control07(UOG_ENDPTCTRL7)	RW
Base + 0x200	Host1	Host 1 ID (UH1_ID)	RO
Base + 0x204	Host1	Hardware General (UH1_HWGENERAL)	RO

Table 60-2. USBOH3 Module Memory Map (continued)

Address	Controller	Use	Access
Base + 0x208	Host1	Host Hardware Parameters (UH1_HWHOST)	RO
Base + 0x210	Host1	Tx Buffer Hardware Parameters (UH1_HWTXBUF)	RO
Base +0x214	Host1	Rx Buffer Hardware Parameters (UH1_HWRXBUF)	RO
Base +0x280	Host1	General Purpose Timer #0 Load(UH1_GPTIMER0LD)	RW
Base +0x284	Host1	General Purpose Timer #0 Controller(UH1_GPTIMER0CTRL)	RW
Base +0x288	Host1	General Purpose Timer #1 Load(UH1_GPTIMER0LD)	RW
Base +0x28c	Host1	General Purpose Timer #1 Controller(UH1_GPTIMER0CTRL)	RW
Base +0x300	Host1	Capability Register Length (UH1_CAPLENGTH)	RO
Base + 0x302	Host1	Host Interface Version (UH1_HCIVERSION)	RO
Base +0x304	Host1	Host Control Structural Parameters (UH1_HCSPARAMS)	RO
Base +0x308	Host1	Control Capability Parameters (UH1_HCCPARAMS)	RO
Base +0x340	Host1	USB Command Register (UH1_USBCMD)	RW
Base +0x344	Host1	USB Status Register (UH1_USBSTS)	RW
Base +0x348	Host1	Interrupt Enable Register (UH1_USBINTR)	RW
Base + 0x34C	Host1	USB Frame Index (UH1_FRINDEX)	RW
Base +0x354	Host1	Host Controller Frame List Base Address (UH1_PERIODICLISTBASE)	RW (32-bit only)
Base +0x358	Host1	Host Controller Next Asynch. Address (UH1_ASYNCLISTADDR)	RW (32-bit only)
Base +0x360	Host1	Host Controller Embedded TT Asynch. Buffer Status (UH1_BURSTSIZE)	RW (32-bit only)
Base +0x364	Host1	TX FIFO Fill Tuning (UH1_TXFILLTUNING)	RW (32-bit only)
Base +0x36C	OTG	IC_USB enable and voltage negotiation(UOG_IC_USB)	RW
Base +0x370	Host1	ULPI Viewport (UH1_ULPIVIEW)	RW
Base +0x380	Host1	Reserved	RO
Base +0x384	Host1	Port Status & Control (UH1_PORTSC1)	RW
Base +0x3A8	Host1	USB Device Mode (UH1_USBMODE)	RW
Base + 0x400	Host2	ID (UH2_ID)	RO
Base + 0x404	Host2	Hardware General (UH2_HWGENERAL)	RO
Base + 0x408	Host2	Host Hardware Parameters (UH2_HWHOST)	RO
Base + 0x410	Host2	TX Buffer Hardware Parameters (UH2_HWTXBUF)	RO
Base +0x414	Host2	RX Buffer Hardware Parameters (UH2_HWRXBUF)	RO
Base +0x480	Host2	General Purpose Timer #0 Load(UH2_GPTIMER0LD)	RW
Base +0x484	Host2	General Purpose Timer #0 Controller(UH2_GPTIMER0CTRL)	RW
Base +0x488	Host2	General Purpose Timer #1 Load(UH2_GPTIMER0LD)	RW
Base +0x48c	Host2	General Purpose Timer #1 Controller(UH2_GPTIMER0CTRL)	RW
Base +0x500	Host2	Capability Register Length (UH2_CAPLENGTH)	RO
Base + 0x502	Host2	Host Interface Version (UH2_HCIVERSION)	RO
Base +0x504	Host2	Host Control Structural Parameters (UH2_HCSPARAMS)	RO
Base +0x508	Host2	Control Capability Parameters (UH2_HCCPARAMS)	RO
Base +0x540	Host2	USB Command Register (UH2_USBCMD)	RW
Base +0x544	Host2	USB Status Register (UH2_USBSTS)	RW
Base +0x548	Host2	Interrupt Enable Register (UH2_USBINTR)	RW

Table 60-2. USBOH3 Module Memory Map (continued)

Address	Controller	Use	Access
Base + 0x54C	Host2	USB Frame Index (UH2_FRINDEX)	RW
Base +0x554	Host2	Host Controller Frame List Base Address (UH2_PERIODICLISTBASE)	RW (32-bit only)
Base +0x558	Host2	Host Controller Next Asynch. Address (UH2_ASYNCLISTADDR)	RW (32-bit only)
Base +0x560	Host2	Host Controller Embedded TT Asynch. Buffer Status (UH2_BURSTSIZE)	RW (32-bit only)
Base +0x564	Host2	TX FIFO Fill Tuning (UH2_TXFILLTUNING)	RW (32-bit only)
Base +0x56C	OTG	IC_USB enable and voltage negotiation(UOG_IC_USB)	RW
Base + 0x570	Host2	ULPI Viewport (UH2_ULPIVIEW)	RW
Base +0x580	Host2	Reserved	RO
Base +0x584	Host2	Port Status & Control (UH2_PORTSC1)	RW
Base +0x5A8	Host2	USB Device Mode (UH2_USBMODE)	RW
Base + 0x600	Host3	ID (UH2_ID)	RO
Base + 0x604	Host3	Hardware General (UH2_HWGENERAL)	RO
Base + 0x608	Host3	Host Hardware Parameters (UH2_HWHOST)	RO
Base + 0x610	Host3	TX Buffer Hardware Parameters (UH2_HWTXBUF)	RO
Base +0x614	Host3	RX Buffer Hardware Parameters (UH2_HWRXBUF)	RO
Base +0x680	Host3	General Purpose Timer #0 Load(UH2_GPTIMER0LD)	RW
Base +0x684	Host3	General Purpose Timer #0 Controller(UH2_GPTIMER0CTRL)	RW
Base +0x688	Host3	General Purpose Timer #1 Load(UH2_GPTIMER0LD)	RW
Base +0x68c	Host3	General Purpose Timer #1 Controller(UH2_GPTIMER0CTRL)	RW
Base +0x700	Host3	Capability Register Length (UH2_CAPLENGTH)	RO
Base + 0x702	Host3	Host Interface Version (UH2_HCIVERSION)	RO
Base +0x704	Host3	Host Control Structural Parameters (UH2_HCSPARAMS)	RO
Base +0x708	Host3	Control Capability Parameters (UH2_HCCPARAMS)	RO
Base +0x740	Host3	USB Command Register (UH2_USBCMD)	RW
Base +0x744	Host3	USB Status Register (UH2_USBSTS)	RW
Base +0x748	Host3	Interrupt Enable Register (UH2_USBINTR)	RW
Base + 0x74C	Host3	USB Frame Index (UH2_FRINDEX)	RW
Base +0x754	Host3	Host Controller Frame List Base Address (UH2_PERIODICLISTBASE)	RW (32-bit only)
Base +0x758	Host3	Host Controller Next Asynch. Address (UH2_ASYNCLISTADDR)	RW (32-bit only)
Base +0x760	Host3	Host Controller Embedded TT Asynch. Buffer Status (UH2_BURSTSIZE)	RW (32-bit only)
Base +0x764	Host3	TX FIFO Fill Tuning (UH2_TXFILLTUNING)	RW (32-bit only)
Base +0x76C	OTG	IC_USB enable and voltage negotiation(UOG_IC_USB)	RW
Base + 0x770	Host3	ULPI Viewport (UH2_ULPIVIEW)	RW
Base +0x780	Host3	Reserved	RO
Base +0x784	Host3	Port Status & Control (UH2_PORTSC1)	RW
Base +0x7A8	Host3	USB Device Mode (UH2_USBMODE)	RW
Base + 0x800	USBOH3	USB Control Register (USB_CTRL)	RW
Base + 0x804	USBOH3	USB OTG Mirror Register (USB_OTG_MIRROR)	RW

Table 60-2. USBOH3 Module Memory Map (continued)

Address	Controller	Use	Access
Base + 0x808	USBOH3	UTMI PHY Control 0 Register (PHY_CTRL_0)	RW
Base + 0x80C	USBOH3	UTMI PHY Control 1 Register (PHY_CTRL_1)	RW
Base + 0x810	USBOH3	USB Control Register (USB_CTRL_1)	RW
Base + 0x814	USBOH3	USB Host2 Control Register (USB_UH2_CTRL)	RW
Base + 0x818	USBOH3	USB Host3 Control Register (USB_UH3_CTRL)	RW

60.2.1 Register Descriptions

This section describes only the registers that are additional to the HS-USB Core register set.

60.2.1.1 USB_CTRL — USB Control Register

The USB control register controls the integration specific features of the USB module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Figure 60-2 shows the USB_CTRL registe.

Base + 0x800

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OWIR	OSIC		OUIE	OWIE	OBPVAL		OPM	ICVOL	Reserved			ICT-PIE	UB-PCKE	H1TCKOEN	ICT-PC
W																
RESET:	-	1	0	0	0	0	0	0	-	1	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	H1WIR	H1SIC		H1UIE	H1WIE	H1BPVAL		H1PM	OHSTLL	H1HSTLL	Re-served	H1DISF-STLL	Reserved		OTCKOEN	BPE
W																
RESET:	-	1	0	0	0	0	0	0	0	0	0	0	00		1	0

Figure 60-2. USB_CTRL Register

Table 60-3 describes the USB_CTRL register fields.

Table 60-3. USB_CTRL Register Field Descriptions

Field	Description
31 OWIR	OTG Wake up Interrupt Request This bit indicates that a wake-up interrupt request is received on the OTG port. This bit is cleared by disabling the wake-up interrupt. 1 Wake-up Interrupt Request received 0 No Wake-up detected
30–29 OSIC	OTG Serial Interface Configuration Controls the interface type of the OTG port when used with a serial transceiver. This bit field allows for configuring the serial interface for single-ended or differential operation combined with bidirectional or unidirectional operation. The available settings are described below. **Note:** When using bidirectional mode, the IOMUX must be configured to place the DP/DM pad into loopback mode. • Bit 30's settings are as follows: 0 Differential/unidirectional (6-wire) 0 Differential/bidirectional (4-wire) 1 Single ended/unidirectional (6-wire) (Default) 1 Single ended/unidirectional (6-wire) (Default) • Bit 29's settings are as follows 0 Differential/unidirectional (6-wire) 1 Differential/bidirectional (4-wire) 0 Single ended/unidirectional (6-wire) (Default) 1 Single ended/unidirectional (6-wire) (Default)
28 OUIE	OTG ULPI interrupt enable Controls whether or not interrupts from the ULPI transceiver will trigger the wake-up logic. This bit is only meaningful when a ULPI transceiver is selected. 1 ULPI transceiver interrupts activate the wake-up logic 0 ULPI transceiver interrupts are ignored by the wakeup logic.
27 OWIE	OTG Wake-up Interrupt Enable This bit enables or disables the OTG wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
26–25 OBPVAL	OTG Bypass Value This field contains the status of the RxDp and RxDm inputs to the OTG core when Bypass mode is enabled. Bit 26 controls RxDp, bit 25 controls RxDm.
24 OPM	OTG Power Mask The power mask bit controls whether or not the external Vbus Power and OverCurrent detection are active for the OTG port. 1 The USBPWR and OC pins are not used by the OTG core. 0 The USBPWR pin will assert with the OTG core's Vbus power Enable and the assertion of the OC input will be reported to the OTG core.
23 ICVOL	Host1 IC-USB voltage status Indicates the voltage status of IC-USB 1 3.0 V class 0 1.8 V class
22–20	Reserved

Table 60-3. USB_CTRL Register Field Descriptions (continued)

Field	Description
19 ICTPIE	IC USB TP interrupt enable 1 IC USB TP interrupt enable 0 IC USB TP interrupt disable
18 UBPCKE	Bypass clock enable 1 Bypass Host1 ulpi clock to OTG port 0 Bypass OTG ulpi clock to Host1 port
17 H1TCKOEN	Host1 ULPI PHY clock output enablej 1 CLK output disable 0 CLK output enable
16 ICTPC	Clear IC TP interrupt flag 1 Clear IC TP interrupt flag 0 Not clear IC TP interrupt
15 H1WIR	Host 1 Wake-up Interrupt Request Indicates a pending Wake-up request on Host port 1. This bit is cleared by disabling the interrupt. The interrupt must be disabled for at least 2 clock cycles of the standby clock. 1 Wake-up interrupt received 0 No Wake-up interrupt received
14–13 H1SIC	Host 1 Serial Interface Configuration Controls the interface type of the Host 1 port when used with a serial transceiver. This bit field allows for configuring the serial interface for single-ended or differential operation combined with bidirectional or unidirectional operation. The available settings are described below. **Note:** In bidirectional mode, IOMUX must set the DP/DM pad into loopback mode. • Bit 14's settings are as follows: 0 Differential/unidirectional (6-wire) 0 Differential/bidirectional (4-wire) 1 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire) (Default) • Bit 13's settings are as follows 0 Differential/unidirectional (6-wire) 1 Differential/bidirectional (4-wire) 0 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire) (Default)
12 H1UIE	Host 1 ULPI interrupt enable Controls whether or not interrupts from the ULPI transceiver will trigger the wake-up logic. This bit is only meaningful when a ULPI transceiver is selected. 1 ULPI transceiver interrupts activate the wake-up logic. 0 ULPI transceiver interrupts are ignored by the wakeup logic.
11 H1WIE	Host 1 Wake-up Interrupt Enable This bit enables or disables the Host 1 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
10–9 H1BPVAL	HOST 1 Bypass Value This field contains the status of the RxDp and RxDm inputs to the HOST1 core when Bypass mode is enabled. Bit 10 controls RxDp, bit 9 controls RxDm.

Table 60-3. USB_CTRL Register Field Descriptions (continued)

Field	Description
8 H1PM	Host 1 Power Mask Used to mask the Host1 USB VBUS enable function. 1 Host 1 VBUS enable is controlled by the Host 1 controller (PORTSC.PP bit) 0 Host 1 VBUS enable is always inactive
7 OHSTLL	OTG ULPI TLL Enable 1 ULPI TLL is Enabled 0 ULPI TLL is Disabled (Default)
6 H1HSTLL	Host 1 ULPI TLL Enable 1 ULPI TLL is Enabled 0 ULPI TLL is Disabled (Default)
5	Reserved
4 H1DISFSTLL	Host 1 Serial TLL disable This bit controls the Serial TLL logic (FS-TLL) for Host Port 1 1 Serial TLL is disabled 0 Serial TLL is enabled (Default)
3–2	Reserved
1 OTCKOEN	OTG ULPI PHY clock output enablej 1 CLK output disable 0 CLK output enable
0 BPE	Bypass Enable This bit enables/disables the USB Bypass function. 1 Bypass active—USB signals from Host port 1 are routed to the OTG port. 0 Bypass inactive—Normal mode operation (Default).

60.2.1.2 USB_OTG_MIRROR—OTG Port Mirror Register

The OTG port is designed for operation with an external OTG transceiver. When a ULPI transceiver is in use, all OTG signaling is communicated over the ULPI data bus as described in the ULPI specification. However, when a serial transceiver is used, the interface for OTG signaling is not standardized. Most OTG transceivers use a serial interface like I^2C or SPI to transfer the OTG signaling back to the CPU and/or USB core. In this case, the USB CORE has no direct connection the OTG signals in the transceiver.

The OTGMIRROR register, shown in Figure 60-3, provides a soft interface between the OTG signals in the transceiver and the OTG signal inputs to the USB core. The USB driver software is responsible for reading the OTG status registers in the transceiver over the serial interface and set the bits accordingly in the OTGMIRROR register (see Figure 60-6), such that the USB controller knows the status of the transceiver.

The USB driver should be designed such that the latency requirements as defined in the USB 2.0 OTG supplement specification are met.

Base + 0x804

	8	7	6	5	4	3	2	1	0
R	UTMIPHY-CLK	ULPIPHY-CLK	Reserved		SESEND	VBUSVAL	BSESVLD	ASESVLD	IDDIG
W									
RESET:	0	0	-	-	0	0	0	0	0

= Unimplemented or Reserved

Figure 60-3. OTG Mirror Register (OTGMIRROR)

Table 60-4 shows the OTG mirror register field descriptions.

Table 60-4. OTG Mirror Register Field Descriptions

Field	Description
8 UTMIPHYCLK	OTG UTMI PHY Clock on detection This is a read-only status bit, indicating the external USB UTMI PHY clock is on and sent to USB controller. 1 The OTG ULPI input clock is on 0 The OTG ULPI input clock is off
7 ULPIPHYCLK	OTG ULPI PHY Clock on detection This is a read-only status bit, indicating the external USB ULPI PHY clock is on and sent to USB controller. 1 The OTG ULPI input clock is on 0 The OTG ULPI input clock is off
6–5	Reserved
4 SESEND	B device Session End This bit is set by the USB driver when the PHY reports a Session End condition. 1 Session end (0.2 V < Vbus < 0.8 V) 0 Session active
3 VBUSVLD	Vbus Valid The USB driver sets this bit when the transceiver reports Vbus Valid. 1 Vbus is valid (Vbus > 4.4 V) 0 Vbus Invalid (Vbus < 4.4 V)
2 BSESVLD	B Session Valid B session valid should be set when the transceiver reports B-session Valid. 1 B Session is valid (0.8 V < Vbus < 4.0 V) 0 B Session is not valid (Vbus < 0.8 V)
1 ASESVLD	A session Valid This bit must be set when a valid ‘A-Session’ level is detected on Vbus. 1 A Session is valid (0.8 V < Vbus < 2.0 V) 0 Session is not valid for A-device.
0 IDIDG	OTG ID-Pin Status This bit indicates to the USB core whether it should operate as A-device or as B-device 1 ID pin is high—Operate as B-device 0 ID pin is low —Operate as A-device

60.2.1.3 PHY_CTRL_0/1—UTMI PHY Control Register 0/1(Base + 0x808/0x80c)

PHY Control Registers 0/1, shown in Figure 60-4 and Figure 60-5, are used to control the internal UTMI PHY.

Base + 0x808

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLOAD	VCONTROL[3:0]				CONF2	CONF3	CHGRDE-TEN	CHGRDETON	VSTATUS[7:1]						
W																
RESET:	1	0000				0	0	0	0	000000						
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VSTATUS[0]	Reserved		SUS-PENDM	RE-SET	UTMI_ON_CLK	OTG_OVRCUR_POL	OTG_OVRCUR_DIS	OTG_XCVR_CLK_SEL	Reserved		H1_XCVR_CLK_SEL	PWR_POL	CHGRDET	CHGRDET_INT_EN	CHGRDET_FLG
W																
RESET:	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0

Figure 60-4. PHY_CTRL_0 Register

Table 60-5 shows the PHY_CTRL_0 field descriptions.

Table 60-5. PHY_CTRL_0 Register Field Descriptions

Field	Description
31 VLOAD	Chipidea UTMI PHY Vload Assertion of this signal loads the Vendor Control register. Active low. 1 Inactive 0 Loads the Vendor Control register
30–27 VCONTROL	Chipidea UTMI PHY Vcontrol
26 CONF2	ChipIdea UTMI CONF2 Active High. When asserted, turns on OTG comparators during suspend (suspend = 0). 1 Turn on OTG comparators during suspend 0 Inactive
25 CONF3	ChipIdea UTMI PHY CONF3 During non-driving mode (opmode[1:0] = 2'b01) if conf3 = 1 the 15 KΩ pull-down resistors are connected (if dppulldown = dnpulldown = '1'). Should be tied to 0 for device only applications.
24 CHGRDETEN	ChipIdea UTMI PHY chgrdeten Active High. Enable Charger Detector. This pin must be asserted 300 µs (or more) after chgrdeton going to '1'.
23 CHGRDETON	ChipIdea UTMI PHY chgrdeton Active High. Charger Detector Power On Control. This pin controls the internal current mirrors used for charger detection. Must be asserted to '1' at least 20 µs before chgrdeten going to 1.
22–15 VSTATUS	ChipIdeat UTMI PHY Vstatus Vendor Status—ChipIdea defined 8-bit parallel output.
14–13	Reserved

Table 60-5. PHY_CTRL_0 Register Field Descriptions (continued)

Field	Description
12 SUSPENDM	UTMI PHY Suspend Active Low. Set PHY into suspend mode 1 Disable 0 Enable
11 Reset	UTMI PHY Reset Active High. Reset the UTMI PHY Reset the PHY 0 Inactive
10 Utmi_on_clock	UTMI PHY On clock Allows the system clocks clockout to be available even if suspend is asserted. 1 Enable 0 Disable
9 OTG_Over_Cur_Pol	OTG Polarity of Overcurrent The polarity of OTG port overcurrent event. 1 Low active 0 High active
8 OTG_Over_Cur_Dis	OTG Disable Overcurrent Event Disable the OTG overcurrent event. 1 Disable the Overcurrent event 0 Enable the overcurrent event
7 OTG_Xcvr_clk_sel	Select the clock source of the xcvr_clk Select the clock source of the xcvr_clk for OTG CORE. 1 ipg_clk_60 Mhz is selected, incase of fslsserialmode for UTMI PHY mode, this bit must be set 0 Depends on which PHY is connected (Serial PHY: ipg_clk_60Mhz; ULPI PHY: ipp_ind_otg_clk; UTMI PHY: sie_clock)
6–5	Reserved
4 H1_Xcvr_clk_sel	Select the clock source of the xcvr_clk Select the clock source of the xcvr_clk for HOST1 CORE. 1 ipg_clk_60 Mhz is selected 0 Depends on which PHY is connected (Serial PHY: ipg_clk_60Mhz; ULPI PHY: ipp_ind_otg_clk; UTMI PHY: sie_clock)
3 PWR_POL	OTG power Pin polarity The polarity of OTG Power pin(to enable external PMIC to drive VBUS) 1 High active 0 Low active
2 CHRGDET	ChipIdea UTMI PHY chrgdet Charger detector output. Note: Maximum response time = 1µs 1 When a charger is detected 0 When a Host is detected

Table 60-5. PHY_CTRL_0 Register Field Descriptions (continued)

Field	Description
1 CHRGDET_INT_EN	chrgdet detected interrupt enable Charger detected interrupt enable. 1 Enable the charger detected interrupt 0 Disable the charger detected interrupt
0 CHRGDET_INT_FLG	chrgdet detected interrupt flag Charger detected interrupt flag. This bit will be cleared when CHRGDET_INT_EN is '0' 1 Charger detected interrupt occurred 2 No charger detected interrupt

Base + 0x80C

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	HS-DRVTIMI-NGP	HSDRVTIM-INGN		HSDRVAM-PLITUDE		HSDRVSLOPE				HSDED-VSEL		HSTEDVSEL		FSTUNEVSEL		
RESET:	0	00		00		0000				10		10		100		

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ICPCTRL		FSRFTSEL		LSRFTSEL		EN-PRE	PREEMDEPTH	CALBP	EXTCAL					PLLDIVVAL-UE	
RESET:	00		01		01		0	0	0	00000					10	

Figure 60-5. PHY_CTRL_1 Register

Table 60-6 shows the PHY_CTRL_1 field descriptions.

Table 60-6. PHY_CTRL_1 Register Field Descriptions

Field	Descriptions
31 hsdrvtimingp	HS driver timing control for PMOS 1 8x 0 2x
30–29 hsdrvtimingn	HS driver timing control for NMOS 00 2x 01 4x 10 6x 11 8x
28–27 hsdrvamplitude	HS driver amplitude control 00 = I (I = 17.78 mA) 01 = I + 2.5% 10 = I + 5% 11 = I + 7.5%

Table 60-6. PHY_CTRL_1 Register Field Descriptions

Field	Descriptions
26–23 hsdrvslope	HS driver slope control The HS Driver may have its rise/fall times controlled using the hsdrvslope[3:0] control pins. Additional charge is injected, so the HS driver rise/fall times change (RC constant changes). Rise/Fall times: Depends on the Package, PCB... Correct value result from Silicon tests
22–21 hsdedvsel	Reference voltage for high speed disconnect envelope detector 00 = (46+2)/100 × vbg (556.8 mV) 01 = (46+3)/100 × vbg (568.4 mV) 10 = (46+4)/100 × vbg (580 mV) 11 = (46+5)/100 × vbg (591.6 mV)
20–19 hstedvsel	Reference voltage for high speed transmission envelope detector 00 = vbg × (9)/100 (104.4 mV) 01 = vbg × (9+1)/100 (116 mV) 10 = vbg × (9+2)/100 (127.6 mV) 11 = vbg × (9+3)/100 (139.2 mV)
18–16 fstunevsel	Reference voltage control for Calibration circuit 000 = (46)/100 × vbg (533.6 mV) 001 = (46+1)/100 × vbg (545.2 mV) 010 = (46+2)/100 × vbg (556.8 mV) 011 = (46+3)/100 × vbg (568.4 mV) 100 = (46+4)/100 × vbg (580 mV) 101 = (46+5)/100 × vbg (591.6 mV) 110 = (46+6)/100 × vbg (603.2 mV) 111 = (46+7)/100 × vbg (614.8 mV)
15–14 icpctrl	PLL charge pump current control 00 = Icp (Icp = 40 µA) 01 = Icp × 0.5 10 = Icp × 1.5 11 = Icp × 2
13–12 fsrftsel	FS driver rise/fall time control 00 = Nominal FS rise time-30% 01 = Nominal FS rise time 10 = Nominal FS rise time 11 = Nominal FS rise time+30%
11–10 lsrftsel	LS driver rise/fall time control 00 = Nominal LS rise time-30% 01 = Nominal LS rise time 10 = Nominal LS rise time 11 = Nominal LS rise time+30%
9 enpre	HS driver pre-emphasis enable
8 preemdepth	HS driver pre-emphasis depth enpre, preemdepth 00 = I (I = 17.78 mA) 01 = I + 5% 10 = I + 10% 11 = I + 20%
7 calbp	Enables calibration bypass for both dp and dn lines

Table 60-6. PHY_CTRL_1 Register Field Descriptions

Field	Descriptions
6–2 extcal	Controls calibration value externally. Valid when calbp = 1
1–0 plldivvalue	Selects between 19.2 MHz, 24 MHz, 26 MHz or 27 MHz reference clock 00 = sysclock uses 19.2 MHz 01 = sysclock uses 24 MHz 10 = sysclock uses 26 MHz 11 = sysclock uses 27 MHz

60.2.1.4 USB_CTRL_1—USB Control Register 1 (Base + 0x810)

Figure 60-6 shows the USB_CTRL_1 register.

Base + 0x80C

	27	26	25	24
R W	uh3_ext_clk_en	uh2_ext_clk_en	uh1_ext_clk_en	otg_ext_clk_en
RESET:	0	0	-	-

Figure 60-6. USB_CTRL_1 Register

Table 60-6 shows the USB_CTRL_1 register field descriptions.

Table 60-7. USB_CTRL_1 Register Field Descriptions

Field	Description
27 uh3_ext_clk_en	Host 3 selecst the clock which comes from external PHY or internal PLL This bit must be set before setting Host 3 core into ULPI mode. It should be cleared when setting Host3 core into FS serial mode. 1 Select the clock which comes from external PHY 0 Select the clock which comes from internal PLL
26 uh2_ext_clk_en	Host 2 selects the clock which comes from external PHY or internal PLL This bit must be set before setting Host 2 core into ULPI mode. It should be cleared when setting Host 2 core into FS serial mode. 1 Select the clock which comes from external PHY 0 Select the clock which comes from internal PLL
25 uh1_ext_clk_en	Host 1 selects whether the clock is from external PHY or internal PLL This bit must be set before setting Host1 core into ULPI mode. It should be cleared when setting Host 1 Core into FS serial mode 1 Select the clock from external PHY 0 Select the clock from internal PLL
24 otg_ext_clk_en	otg selects whether the clock comes from external PHY or internal PLL This bit must be set before setting OTG core into ULPI mode. It should be cleared when setting OTG Core into FSserial mode. 1 Select the clock from external PHY 0 Select the clock from internal PLL

60.2.1.5 USB_UH2_CTRL—USB Host2 Control Register

The USB Host2 control register, shown in Figure 60-7, controls the integration specific features of the USB host2 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Base + 0x814

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												OTG_SER_DRVEN	H1_SER_DRVEN	ICVOL	H2WIR	Reserved
W																
RESET:	-	1	0	0	0	0	0	0	-	1	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		ICT-PIE	H2TCKKOEN	ICT-PC	H2SIC		H2UIE	H2WIE	OVBWK_EN	OIDWK_EN	H2PM	H2HSTLL	Reserved	H2DISF-STLL	Reserved
W																
RESET:	00		0	1	0	1	0	0	0	0	0	0	00		1	0

Figure 60-7. USB_UH2_CTRL Register

Table 60-8 describes the USB Host2 control register's fields.

Table 60-8. USB_UH2_CTRL Register Field Descriptions

Field	Description
31–21	Reserved
20 OTG_SER_DRVEN	OTG Serial interface drive enable OTG core serial interface output enable. By default all serial interface output pins are not driven by the controller. This bit should be enabled after OTG core switches to serial mode and should be disabled before OTG core switches to ULPI or UTMI mode. 1 Serial interface drive enable 0 Serial interface drive disable (default)
19 H1_SER_DRVEN	Host1 Serial interface drive enable Host1 core serial interface output enable. By default all serial interface output pins are not driven by the controller. This bit should be enabled after Host1 core switches to serial mode and should be disabled before Host1 core switches to ULPI mode. 1 Serial interface drive enable 0 Serial interface drive disable (default)
18 ICVOL	Host1 IC-USB voltage status Indicates the voltage status of IC-USB 1 3.0 V class 0 1.8 V class
17 H2WIR	Host 2 Wake-up Interrupt Request Indicates a pending wake-up request on host port 2. This bit is cleared by disabling the interrupt. The interrupt must be disabled for at least 2 clock cycles of the standby clock. 1 Wake-up interrupt received 0 No Wake-up interrupt received
16–14	Reserved

Table 60-8. USB_UH2_CTRL Register Field Descriptions (continued)

Field	Description
13 ICTPIE	IC USB TP interrupt enable 1 IC USB TP interrupt enable 0 IC USB TP interrupt disable
12 H2TCKOEN	Host2 ULPI PHY clock output enable 1 CLK output disable 0 CLK output enable
11 ICTPC	IC USB TP interrupt clear 1 Clear IC USB TP interrupt 0 Not clear
10–9 H2SIC	Host 2 Serial Interface Configuration Controls the interface type of the Host 2 port when used with a serial transceiver. This bit field allows the configuration of the serial interface for single ended or differential operation combined with bidirectional or unidirectional operation. Settings are as shown below. **Note:** In bidirectional mode, IOMUX must set the DP/DM pad into loopback mode. • Bit 14's settings are as follows: 0 Differential/unidirectional (6-wire) 0 Differential/bidirectional (4-wire) 1 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire) • Bit 13's settings are as follows 0 Differential/unidirectional (6-wire) 1 Differential/bidirectional (4-wire) 0 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire)
8 H2UIE	Host 2 ULPI interrupt enable Controls whether or not interrupts from the ULPI transceiver will trigger the wake-up logic. This bit is only meaningful when a ULPI transceiver is selected. 1 ULPI transceiver interrupts activate the wake-up logic 0 ULPI transceiver interrupts are ignored by the wakeup logic.
7 H2WIE	Host 2 Wake-up Interrupt Enable This bit enables or disables the Host 2 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
6 OVBWK_EN	OTG VBUS Wakeup Enable This bit enables or disables the interrupt which is caused OTG VBUS Change. 1 Interrupt Enabled 0 Interrupt Disabled
5 OIDWK_EN	OTG ID Wakeup Enable This bit enables or disables the interrupt which is caused OTG ID Change. 1 Interrupt Enabled 0 Interrupt Disabled

Table 60-8. USB_UH2_CTRL Register Field Descriptions (continued)

Field	Description
4 H2PM	Host 2 Power Mask The power mask bit controls whether or not the external Vbus Power and OverCurrent detection are active for the Host 1 port. 1 The USBPWR and OC pins are not used by the Host 2 core. 0 The USBPWR pin will assert with the Host 2core's Vbus power Enable and the assertion of the OC input will be reported to the Host 2 core.
3 H2HSTLL	Host2 ULPI TLL Enable 1 ULPI TLL is Enabled 0 ULPI TLL is Disabled (Default)
2	Reserved
1 H2DISFSTLL	Host 2Serial TLL disable This bit controls the Serial TLL logic (FS-TLL) for Host Port 2 1 Serial TLL is disabled 0 Serial TLL is enabled (Default)
0	Reserved

60.2.1.6 USB_UH3_CTRL—USB Host3 Control Register

The USB Host3 control register, shown in Figure 60-8, controls the integration specific features of the USB host3 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Base + 0x818

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R												H2_SER_DRVEN	H3_SER_DRVEN	ICVOL	H2WIR	Reserved
W																
RESET:	-	1	0	0	0	0	0	0	-	1	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		ICT-PIE	H3TCKKOEN	ICT-PC	H3SIC		H3UIE	H3WIE	H3BPVAL		H3PM	H3HSTLL	Reserved	H3DISF-STLL	Reserved
W																
RESET:	00		0	1	0	1	0	0	0	0	0	0	00		1	0

Figure 60-8. USB_UH3_CTRL Register

Table 60-8 describes the USB Host3 control register's fields.

Table 60-9. USB_UH3_CTRL Register Field Descriptions

Field	Description
31–21	Reserved
20 H2_SER_DRVEN	Host2 Serial interface drive enable Host2 core serial interface output enable. By default all serial interface output pins are not driven by the controller. This bit should be enabled after Host2 core switches to serial mode and should be disabled before Host2 core switches to ULPI or UTMI mode. 1 Serial interface drive enable 0 Serial interface drive disable (default)
19 H3_SER_DRVEN	Host3 Serial interface drive enable Host3 core serial interface output enable. By default all serial interface output pins are not driven by the controller. This bit should be enabled after Host3 core switches to serial mode and should be disabled before Host3 core switches to ULPI mode. 1 Serial interface drive enable 0 Serial interface drive disable (default)
18 ICVOL	Host1 IC-USB voltage status Indicates the voltage status of IC-USB 1 3.0 V class 0 1.8 V class
17 H3WIR	Host 3 Wake-up Interrupt Request Indicates a pending wake-up request on Host port 3. This bit is cleared by disabling the interrupt. The interrupt must be disabled for at least 2 clock cycles of the standby clock. 1 Wake-up interrupt received 0 No Wake-up interrupt received
16–14	Reserved
13 ICTPIE	IC USB TP interrupt enable 1 IC USB TP interrupt enable 0 IC USB TP interrupt disable
12 H3TCKOEN	Host3 ULPI PHY clock output enable 1 CLK output disable 0 CLK output enable
11 ICTPC	IC USB TP interrupt clear 1Clear IC USB TP interrupt 0not clear
10–9 H3SIC	Host 1 Serial Interface Configuration Controls the interface type of the Host 3 port when used with a serial transceiver. This bit field allows for configuring the serial interface for single-ended or differential operation combined with bidirectional or unidirectional operation. The available settings are described below. **Note:** In bidirectional mode, IOMUX must set the DP/DM pad into loopback mode. • Bit 10's settings are as follows: 0 Differential/unidirectional (6-wire) 0 Differential/bidirectional (4-wire) 1 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire) • Bit 9's settings are as follows 0 Differential/unidirectional (6-wire) 1 Differential/bidirectional (4-wire) 0 Single ended/unidirectional (6-wire) (Default) 1 Single ended/biidirectional (3-wire)

Table 60-9. USB_UH3_CTRL Register Field Descriptions (continued)

Field	Description
8 H3UIE	Host 3 ULPI interrupt enable Controls whether or not interrupts from the ULPI transceiver will trigger the wake-up logic. This bit is only meaningful when a ULPI transceiver is selected. 1 ULPI transceiver interrupts activate the wake-up logic 0 ULPI transceiver interrupts are ignored by the wakeup logic.
7 H3WIE	Host 3 Wake-up Interrupt Enable This bit enables or disables the Host 3 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode. 1 Interrupt Enabled 0 Interrupt Disabled
6–5 H3BPVAL	HOST 3 Bypass Value This field contains the status of the RxDp and RxDm inputs to the HOST3 core when Bypass mode is enabled. Bit 6 controls RxDp, and bit 5 controls RxDm.
4 H3PM	Host 3 Power Mask The power mask bit controls whether or not the external Vbus Power and Over Current detection are active for the Host 3 port. 1 The USBPWR and OC pins are not used by the Host 3 core. 0 The USBPWR pin asserts with the Host 1core's Vbus power Enable and the assertion of the OC input is reported to the Host 3 core.
3 H3HSTLL	Host3 ULPI TLL Enable 1 ULPI TLL is Enabled 0 ULPI TLL is Disabled (Default)
2	Reserved
1 H3DISFSTLL	Host 3 Serial TLL disable This bit controls the Serial TLL logic (FS-TLL) for Host Port 3 1 Serial TLL is disabled 0 Serial TLL is enabled (Default)
0	Reserved

60.2.1.7 USB Core Register

For detailed descriptions of the OTG Controller, Host 1, and Host 2 Controller registers, refer to Section 60.4.2, Register Interface." The value of registers that depend on implementation can be found in the core configuration parameter file (for the OTG Controller refer to Section 60.3.4.1, OTG Controller Configuration Parameter," for the Host1 Controller refer to Section 60.3.1.1, Host Controller 1 Configuration Parameter," and for the Host2 Controller refer to Section 60.3.2.1, Host Controller 2 Configuration Parameter."

60.3 Functional Description

This section describes the functionality and the topology of the different building blocks of the USB module.

60.3.1 USB HOST Controller 1

Host Controller 1 is an instantiation of the core. The core is configured for high-speed/full-speed/low-speed operation.

60.3.1.1 Host Controller 1 Configuration Parameter

Host controller 1 is configured for high-speed/full-speed/low-speed operation and supports serial transceiver interface and ULPI interface. The parameters in Example 60-1 are set for the implementation of the host controller core.

Example 60-1. Host Controller Implementation

```
// ------------------------------------------------------------------------
// ------------------------------------------------------------------------
//  System Configuration Options
// ------------------------------------------------------------------------
//
//  The VUSB_HS_RESET_TYPE constant determines the reset type used in the core.
//    0 = Use Synchronous Resets
//    1 = Use Asynchronous Resets
//
//  The VUSB_HS_RESET_OPTIONAL constant determines if various register files
//  are connected to the main reset.
//
//  0 = Reset all flops except those listed above.
//  1 = Reset 100% of flops.
//
parameter VUSB_HS_RESET_TYPE = 1; //  [0,1]
parameter VUSB_HS_RESET_OPTIONAL = 0; //  [0,1]
//
//  The VUSB_HS_CLOCK_CONFIGURATION constant determines the clocking used in the core
//  See reference manual for a description of the clocking modes available.
//
//    0 = xcvr_clk_0 = pe_clk = clk (do not select for multi-port host product)
//    1 = xcvr_clk_0 < pe_clk = clk
//    2 = xcvr_clk_0 = pe_clk <> clk (do not select for multi-port host product)
//    3 = xcvr_clk_0 < pe_clk <> clk
parameter VUSB_HS_CLOCK_CONFIGURATION = 2; //  [0,1,2,3]
//
//  phy size (UTMI Only)
//  Set according to the data width of the transciever connected to the core.
//    0 = 8 bit wide data bus [60MHz clock from the transciever]
//    1 = 16 bit wide data bus [30MHZ clock from the transciever]
//    2 = software programmable reset to 8-bit width
//    3 = software programmable reset to 16-bit width
//  NOTE: options 2,3 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
parameter VUSB_HS_PHY16_8 = 0; //  [0,1,2,3]
//  This constant selects which phy is being used
//  0 = UTMI/UMTI+
```

```
//  1 = Philips legacy style
//  2 = ULPI ** Must have purchased the ULPI Option **
//  3 = Serial Only
//  4 = Software programmable - reset to UTMI
//  5 = Software programmable - reset to Philips legacy style
//  6 = Software programmable - reset to ULPI ** Must have purchased the ULPI Option **
//  7 = Software programmable - reset to Serial
//  NOTE: options 4,5,6,7 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
//  NOTE: undefined results if ULPI is selected (#2,6) and the ULPI option was not purchased &
installed.
parameter VUSB_HS_PHY_TYPE = 7; //  [0,1,2,3,4,5,6,7]
//  derived constants ; must set to match VUSB_HS_PHY_TYPE above according the attached
instructions.
parameter VUSB_HS_PHY_UTMI = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=1,2,3       else 1
parameter VUSB_HS_PHY_PHIL = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,2,3       else 1
parameter VUSB_HS_PHY_ULPI = 1; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,1,3       else 1
//  This constant selects that the serial engine is used in place of the
//  parallel signalling for UTMI+,Philips, and ULPI.
//  0 - No Serial Engine      - Always use parallel signalling
//  1 - Serial Engine Present - Always use serial signalling for FS/LS
//  2 - Software programmable - Reset to use parallel signalling for FS/LS
//  3 - Software programmable - Reset to use serial signalling for FS/LS
//  NOTE: Must set to 1 if PHY_TYPE=1 or 3
//  NOTE: Must set to 1 or 3 is PHY_TYPE=6
//  NOTE: option 0 slims gate count by removing all logic necessary in the serial engine
parameter VUSB_HS_PHY_SERIAL = 2;
// ------------------------------------------------------------------------
//  Device Configuration
// ------------------------------------------------------------------------
//
//  The VUSB_HS_DEV_EP constant represents the total number of endpoints.
//
//  ** NOTE ** : VUSB_HS_DEV_EP should be set to 1 for host only implementations,
//  ** NOTE ** : VUSB_HS_DEV_EP_ADD must be updated when the VUSB_HS_DEV_EP is changed.
//
parameter VUSB_HS_DEV_EP = 1; //  [2,3,...,16]
//  set the VUSB_HS_DEV_EP_ADD equal to max(1, log2(VUSB_HS_DEV_EP))
parameter VUSB_HS_DEV_EP_ADD = 1;
// ------------------------------------------------------------------------
//  Host Configuration
// ------------------------------------------------------------------------
//  The VUSB_HS_NUM_PORT constant specifies the number of downstream ports
//  supported by the host controller.  This constant is only relevant for
//  for the multi-port host product.  Leave 1 for device, otg, and single port host
//  products.
//
//  Maximum: 8 downstream ports.
//
parameter VUSB_HS_NUM_PORT = 1; //  [1,2,...,8]
// ------------------------------------------------------------------------
//  Host Configuration - Internal Transaction Translator
// ------------------------------------------------------------------------
//  These constants define how many periodic contexts the internal TT can support.
//
//  Note1: These constants are only relevent for the multi-port host product.
//  Note2: The number of async. contexts should not be adjusted.
//
```

```
//  USB 2.0 spec requires 16 periodic contexts for a hub TT although this seems unncessarily
//  big for an embedded application where limits can be imposed on the downstream applications.
//  Changing the  number from 16 to 4 will significantly reduce gate count but be cautioned
//  that this will impose limits on the number of downstream periodic (ISO/Interrupt)
transactions.
//
//  The simplifying limit imposed when the number of periodic contexts is reduced to 4 is
//  that the application can send no more than 4 periodic (ISO/Interrupt) packets per
//  frame to the downstream Full and Low-Speed devices. Whereas, when the number
//  of periodic contexts are 16, its possible to pipeline packets accross the
//  microframes and send as many periodic packets per frame as can be scheduled (> 16).
//
parameter VUSB_HS_TT_ASYNC_CONTEXTS = 2; //  DO NOT CHANGE
parameter VUSB_HS_TT_ASYNC_CONTEXTS_ADDR = 1; //  DO NOT CHANGE ;
max(1,log2(VUSB_HS_TT_ASYNC_CONTEXTS))
parameter VUSB_HS_TT_PERIODIC_CONTEXTS = 16; //  [4,16 Only] ; Caution (see note above)
//  if this is changed to 4 then change the following constants:
//     VUSB_HS_TT_PERIODIC_CONTEXT_ADDR=2
//     VUSB_HS_TT_CONTEXTS_ADDR=2
parameter VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR = 4; //  max(1,log2(VUSB_HS_TT_PERIODIC_CONTEXTS))
parameter VUSB_HS_TT_CONTEXTS_ADDR = 4; //  max(VUSB_HS_TT_ASYNC_CONTEXTS_ADDR,
VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR)
// ------------------------------------------------------------------------
//  RX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For RX Buffer
//  ** NOTE ** : VUSB_HS_RX_ADD must be updated when VUSB_HS_RX_DEPTH is changed.
parameter VUSB_HS_RX_DEPTH = 128; //  Must be power of 2! (ie. 2**j)
//  Minimum : >= 8
//  Minimum : >= 3*VUSB_HS_RX_BURST/2
//  Maximum : <= 2048
//  Recommended : > 2*VUSB_HS_RX_BURST
parameter VUSB_HS_RX_ADD = 7; //  log2(VUSB_HS_RX_DEPTH)
parameter VUSB_HS_RX_BURST = 8; //  Burst size for RX Buffer To Memory Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
// ------------------------------------------------------------------------
//  TX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For TX Channel
//  The Channel constants control the size of the buffer associated with each TX
//  endpoint channel.  All of the channel buffers are combined into a single
//  TX Buffer using the TX Buffer constants.
//  ** NOTE ** : VUSB_HS_TX_CHAN_ADD must be updated when VUSB_HS_TX_CHAN is changed.
parameter VUSB_HS_TX_CHAN = 128; //  Must be power of 2 (ie. 2**k)
//  Minimum : >= 16
//  Minimum : >= 3*VUSB_HS_TX_BURST/2+4
//  Maximum : <= 128
//  Recommended : > 2*VUSB_HS_TX_BURST+4
parameter VUSB_HS_TX_CHAN_ADD = 7; //  log2(VUSB_HS_TX_CHAN);
parameter VUSB_HS_TX_CHAN_STATE = 4; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_STATE_ADD = 2; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_DATA = VUSB_HS_TX_CHAN - VUSB_HS_TX_CHAN_STATE; //  DO NOT CHANGE
//  Depth & Number Of Address Bits For TX Buffer
//  ** NOTE ** : VUSB_HS_TX_DEPTH and VUSB_HS_TX_ADD must be updated when either
//               VUSB_HS_TX_CHAN or VUSB_HS_DEV_EP are changed.
```

```
//  Host Only : VUSB_HS_TX_DEPTH = VUSB_HS_TX_CHAN
//  Device    : VUSB_HS_TX_DEPTH = (VUSB_HS_TX_CHAN)*min(1,VUSB_DEV_EP)
//          : VUSB_HS_TX_DEPTH <--> ** Power of 2 not necessary ** ; integer multiple of power
of 2 acceptable;
//                                    ie. 2**X * VUSB_HS_DEV_EP.
//                              (verify code exists for chosen TX_DEPTH in graycode package file)
parameter VUSB_HS_TX_DEPTH = 128; //  (VUSB_HS_TX_CHAN)*max_int(1,VUSB_HS_DEV_EP);
parameter VUSB_HS_TX_ADD = 7; //  log2(VUSB_HS_TX_DEPTH);
parameter VUSB_HS_TX_BURST = 8; //  Burst size for Memory To TX Buffer Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
//
//  Device/OTG : This constant determines if the device endpoint
//  context data used for rapid transmit response (reduced latency) and
//  double buffering tracking is stored in the TX-FIFO or if it is held
//  in registers.  There are advantages and disadvantages to each configuration:
//
//  = 0 (context in FIFO)
//     advantage    : smaller gate count & power (less 126 registers + misc. per endpoint)
//    disadvantage : actual latency buffer size is smaller because 4 words are used for context
(ie. actual VUSB_HS_TX_CHAN-4 words)
//                   FIFO interface is 2 read ports / 1 write port***
//
//  = 1 (context is registers)
//     advantage    : latency buffer uses full VUSB_HS_TX_CHAN words for data
//                   FIFO interface is 1 read port / 1 write port***
//     disadvantage : larger gate count & power (adds 144 registers + misc.)
//
//  *** this difference MAY be significant with some technologies and MAY offset the gate count
and power difference.
//
parameter VUSB_HS_TX_LOCAL_CONTEXT_REGISTERS = 1; //  [0 - context in FIFO, 1 - context in
registers]
// ------------------------------------------------------------------------
//  Device Version Number
// ------------------------------------------------------------------------
parameter VUSB_HS_DCIVERSION = 16'b 0000000000000001;
parameter VUSB_HS_HCIVERSION = 16'b 0000000100000000; //  EHCI 1.00
```

60.3.1.2 Host Controller 1 to Host Port 1 Interface

Host 1 core USB signals do not connect directly to the HOST1 I/O pins. Instead, the signals pass through the bypass mux and serial and ULPI mux, which allows additional functionality on Host Port1. The module defaults to serial mode. See Section 60.3.7, USB Bypass Mode," for Bypass Mux details.

Table 60-10 details the I/O pad connections for the USB Host Port 1. The direction control is logic 1 for output.

Table 60-10. Host Port 1 Signal Connections

I/O PAD	Type	Input	Output	Direction Control
USB1_ULPI_CLK	I/O	ipp_ind_uh1_clk	ipp_do_uh1_clk	ipp_obe_uh1_clk
USB1_ULPI_DIR	I/O	ipp_ind_uh1_dir	ipp_do_uh1_dir	ipp_obe_uh1_dir

Table 60-10. Host Port 1 Signal Connections (continued)

I/O PAD	Type	Input	Output	Direction Control
USB1_ULPI_STP	I/O	ipp_ind_uh1_stp	ipp_do_uh1_stp	ipp_obe_uh1_stp
USB1_ULPI_NXT	I/O	ipp_ind_uh1_nxt	ipp_do_uh1_nxt	ipp_obe_uh1_nxt
USB1_ULPI_DATA0	I/O	ipp_ind_uh1_data0	ipp_do_uh1_data0	ipp_obe_uh1_data0
USB1_ULPI_DATA1	I/O	ipp_ind_uh1_data1	ipp_do_uh1_data1	ipp_obe_uh1_data1
USB1_ULPI_DATA2	I/O	ipp_ind_uh1_data2	ipp_do_uh1_data2	ipp_obe_uh1_data2
USB1_ULPI_DATA3	I/O	ipp_ind_uh1_data3	ipp_do_uh1_data3	ipp_obe_uh1_data3
USB1_ULPI_DATA4	I/O	ipp_ind_uh1_data4	ipp_do_uh1_data4	ipp_obe_uh1_data4
USB1_ULPI_DATA5	I/O	ipp_ind_uh1_data5	ipp_do_uh1_data5	ipp_obe_uh1_data5
USB1_ULPI_DATA6	I/O	ipp_ind_uh1_data6	ipp_do_uh1_data6	ipp_obe_uh1_data6
USB1_ULPI_DATA7	I/O	ipp_ind_uh1_data7	ipp_do_uh1_data7	ipp_obe_uh1_data7

Host port 1can support either a ULPI transceiver or a Serial Transceiver (FS/LS). Depending on the selected type, signaling for one or the other is available at the top level. Table 60-13 gives the relation between the top level signal and the USB core signal in both modes.

Table 60-11. / Serial Muxing

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	Serial Mode	ULPI Mode	To I/O Mux	Serial Mode	ULPI Mode
ipp_ind_uh1_data0	ipp_ind_uh1_oe_n	uh1_data_in[0]	ipp_do_uh1_data0	ipp_do_uh1_oe_n	uh1_data_out[0]
ipp_ind_uh1_data1	ipp_ind_uh1_txdm_rxvm_se0	uh1_data_in[1]	ipp_do_uh1_data1	ipp_do_uh1_txdp_dat	uh1_data_out[1]
ipp_ind_uh1_data2	ipp_ind_uh1_rxvm_txdm_se0	uh1_data_in[2]	ipp_do_uh1_data2	ipp_do_uh1_txdm_se0	uh1_data_out[2]
ipp_ind_uh1_data3	ipp_ind_uh1_txdp_rxvp_dat	uh1_data_in[3]	ipp_do_uh1_data3	ipp_do_uh1_suspend	uh1_data_out[3]
ipp_ind_uh1_data4	ipp_ind_uh1_txdp_rxvp_dat	uh1_data_in[4]	ipp_do_uh1_data4	ipp_do_uh1_rxvp_txdp_dat	uh1_data_out[4]
ipp_ind_uh1_data5	ipp_ind_uh1_rxvm_txdm_se0	uh1_data_in[5]	ipp_do_uh1_data5	ipp_do_uh1_rxvm_txdm_se0	uh1_data_out[5]
ipp_ind_uh1_data6	ipp_ind_uh1_xcvr_ser_rcv	uh1_data_in[6]	ipp_do_uh1_data6	ipp_do_uh1_xcvr_ser_rcv	uh1_data_out[6]
ipp_ind_uh1_data7	—	uh1_data_in[7]	ipp_do_uh1_data7	ipp_do_uh1_speed	uh1_data_out[7]
			USB Port 1 Direction Control		
—	—	—	ipp_obe_uh1_data0	ipp_obe_uh1_oe_n	uh1_data_out_en

Table 60-11. / Serial Muxing (continued)

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	Serial Mode	ULPI Mode	To I/O Mux	Serial Mode	ULPI Mode
—	—	—	ipp_obe_uh1_data1	~uh1_oe_n	uh1_data_out_en
—	—	—	ipp_obe_uh1_data2	~uh1_oe_n	uh1_data_out_en
—	—	—	ipp_obe_uh1_data3	1’b0	uh1_data_out_en
—	—	—	ipp_obe_uh1_data4	ipp_obe_uh1_rxvp_txdp_dat	uh1_data_out_en
—	—	—	ipp_obe_uh1_data5	ipp_obe_uh1_rxvm_txdm_se0	uh1_data_out_en
—	—	—	ipp_obe_uh1_data6	ipp_obe_uh1_xcvr_ser_rcv	uh1_data_out_en
—	—	—	ipp_obe_uh1_data7	1'b0	uh1_data_out_en

60.3.2 USB Host Controller 2

The host 2 core USB signals do not connect directly to the HOST2 I/O pins. Instead, the signals pass through the HS-ULPI TLL and serial and ULPI mux, which allows additional functionality on Host Port2. Host controller 2 is configured for operation with a FS/LS serial transceiver or a parallel ULPI transceiver (HS/FS/LS). The selection between transceiver type is software programmable. The module defaults to serial mode.

60.3.2.1 Host Controller 2 Configuration Parameter

Host Controller 2 is configured for high-speed/full-speed/low-speed operation and supports serial transceiver interface and ULPI interface. Example 60-2 shows sample code used to set the host controller core implementation.

Example 60-2. Host Controller 2 Core Implementation

```
// ------------------------------------------------------------------------
// ------------------------------------------------------------------------
//  System Configuration Options
// ------------------------------------------------------------------------
//
//  The VUSB_HS_RESET_TYPE constant determines the reset type used in the core.
//    0 = Use Synchronous Resets
//    1 = Use Asynchronous Resets
//
//  The VUSB_HS_RESET_OPTIONAL constant determines if various register files
//  are connected to the main reset.
//
//  0 = Reset all flops except those listed above.
//  1 = Reset 100% of flops.
```

```
//
parameter VUSB_HS_RESET_TYPE = 1; //  [0,1]
parameter VUSB_HS_RESET_OPTIONAL = 0; //  [0,1]
//
//  The VUSB_HS_CLOCK_CONFIGURATION constant determines the clocking used in the core
//  See reference manual for a description of the clocking modes available.
//
//    0 = xcvr_clk_0 = pe_clk = clk (do not select for multi-port host product)
//    1 = xcvr_clk_0 < pe_clk = clk
//    2 = xcvr_clk_0 = pe_clk <> clk (do not select for multi-port host product)
//    3 = xcvr_clk_0 < pe_clk <> clk
parameter VUSB_HS_CLOCK_CONFIGURATION = 2; //  [0,1,2,3]
//
//  phy size (UTMI Only)
//  Set according to the data width of the transciever connected to the core.
//    0 = 8 bit wide data bus [60MHz clock from the transciever]
//    1 = 16 bit wide data bus [30MHZ clock from the transciever]
//    2 = software programmable reset to 8-bit width
//    3 = software programmable reset to 16-bit width
//  NOTE: options 2,3 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
parameter VUSB_HS_PHY16_8 = 0; //  [0,1,2,3]
//  This constant selects which phy is being used
//  0 = UTMI/UMTI+
//  1 = Philips legacy style
//  2 = ULPI ** Must have purchased the ULPI Option **
//  3 = Serial Only
//  4 = Software programmable - reset to UTMI
//  5 = Software programmable - reset to Philips legacy style
//  6 = Software programmable - reset to ULPI ** Must have purchased the ULPI Option **
//  7 = Software programmable - reset to Serial
//  NOTE: options 4,5,6,7 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
//  NOTE: undefined results if ULPI is selected (#2,6) and the ULPI option was not purchased &
installed.
parameter VUSB_HS_PHY_TYPE = 7; //  [0,1,2,3,4,5,6,7]
//  derived constants ; must set to match VUSB_HS_PHY_TYPE above according the attached
instructions.
parameter VUSB_HS_PHY_UTMI = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=1,2,3       else 1
parameter VUSB_HS_PHY_PHIL = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,2,3       else 1
parameter VUSB_HS_PHY_ULPI = 1; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,1,3       else 1
//  This constant selects that the serial engine is used in place of the
//  parallel signalling for UTMI+,Philips, and ULPI.
//  0 - No Serial Engine      - Always use parallel signalling
//  1 - Serial Engine Present - Always use serial signalling for FS/LS
//  2 - Software programmable - Reset to use parallel signalling for FS/LS
//  3 - Software programmable - Reset to use serial signalling for FS/LS
//  NOTE: Must set to 1 if PHY_TYPE=1 or 3
//  NOTE: Must set to 1 or 3 is PHY_TYPE=6
//  NOTE: option 0 slims gate count by removing all logic necessary in the serial engine
parameter VUSB_HS_PHY_SERIAL = 2;
// ------------------------------------------------------------------------
//  Device Configuration
// ------------------------------------------------------------------------
//
//  The VUSB_HS_DEV_EP constant represents the total number of endpoints.
//
//  ** NOTE ** : VUSB_HS_DEV_EP should be set to 1 for host only implementations,
//  ** NOTE ** : VUSB_HS_DEV_EP_ADD must be updated when the VUSB_HS_DEV_EP is changed.
```

```
//
parameter VUSB_HS_DEV_EP = 1; //  [2,3,...,16]
//  set the VUSB_HS_DEV_EP_ADD equal to max(1, log2(VUSB_HS_DEV_EP))
parameter VUSB_HS_DEV_EP_ADD = 1;
// ------------------------------------------------------------------------
//  Host Configuration
// ------------------------------------------------------------------------
//  The VUSB_HS_NUM_PORT constant specifies the number of downstream ports
//  supported by the host controller.  This constant is only relevant for
//  for the multi-port host product.  Leave 1 for device, otg, and single port host
//  products.
//
//  Maximum: 8 downstream ports.
//
parameter VUSB_HS_NUM_PORT = 1; //  [1,2,...,8]
// ------------------------------------------------------------------------
//  Host Configuration - Internal Transaction Translator
// ------------------------------------------------------------------------
//  These constants define how many periodic contexts the internal TT can support.
//
//  Note1: These constants are only relevent for the multi-port host product.
//  Note2: The number of async. contexts should not be adjusted.
//
//  USB 2.0 spec requires 16 periodic contexts for a hub TT although this seems unncessarily
//  big for an embedded application where limits can be imposed on the downstream applications.
//  Changing the  number from 16 to 4 will significantly reduce gate count but be cautioned
//  that this will impose limits on the number of downstream periodic (ISO/Interrupt)
transactions.
//
//  The simplifying limit imposed when the number of periodic contexts is reduced to 4 is
//  that the application can send no more than 4 periodic (ISO/Interrupt) packets per
//  frame to the downstream Full and Low-Speed devices. Whereas, when the number
//  of periodic contexts are 16, its possible to pipeline packets accross the
//  microframes and send as many periodic packets per frame as can be scheduled (> 16).
//
parameter VUSB_HS_TT_ASYNC_CONTEXTS = 2; //  DO NOT CHANGE
parameter VUSB_HS_TT_ASYNC_CONTEXTS_ADDR = 1; //  DO NOT CHANGE ;
max(1,log2(VUSB_HS_TT_ASYNC_CONTEXTS))
parameter VUSB_HS_TT_PERIODIC_CONTEXTS = 16; //  [4,16 Only] ; Caution (see note above)
//  if this is changed to 4 then change the following constants:
//     VUSB_HS_TT_PERIODIC_CONTEXT_ADDR=2
//     VUSB_HS_TT_CONTEXTS_ADDR=2
parameter VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR = 4; //  max(1,log2(VUSB_HS_TT_PERIODIC_CONTEXTS))
parameter VUSB_HS_TT_CONTEXTS_ADDR = 4; //  max(VUSB_HS_TT_ASYNC_CONTEXTS_ADDR,
VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR)
// ------------------------------------------------------------------------
//  RX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For RX Buffer
//  ** NOTE ** : VUSB_HS_RX_ADD must be updated when VUSB_HS_RX_DEPTH is changed.
parameter VUSB_HS_RX_DEPTH = 128; //  Must be power of 2! (ie. 2**j)
//  Minimum : >= 8
//  Minimum : >= 3*VUSB_HS_RX_BURST/2
//  Maximum : <= 2048
//  Recommended : > 2*VUSB_HS_RX_BURST
parameter VUSB_HS_RX_ADD = 7; //  log2(VUSB_HS_RX_DEPTH)
parameter VUSB_HS_RX_BURST = 8; //  Burst size for RX Buffer To Memory Transfers
```

```
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
// ------------------------------------------------------------------------
//  TX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For TX Channel
//  The Channel constants control the size of the buffer associated with each TX
//  endpoint channel.  All of the channel buffers are combined into a single
//  TX Buffer using the TX Buffer constants.
//  ** NOTE ** : VUSB_HS_TX_CHAN_ADD must be updated when VUSB_HS_TX_CHAN is changed.
parameter VUSB_HS_TX_CHAN = 128; //  Must be power of 2 (ie. 2**k)
//  Minimum : >= 16
//  Minimum : >= 3*VUSB_HS_TX_BURST/2+4
//  Maximum : <= 128
//  Recommended : > 2*VUSB_HS_TX_BURST+4
parameter VUSB_HS_TX_CHAN_ADD = 7; //  log2(VUSB_HS_TX_CHAN);
parameter VUSB_HS_TX_CHAN_STATE = 4; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_STATE_ADD = 2; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_DATA = VUSB_HS_TX_CHAN - VUSB_HS_TX_CHAN_STATE; //  DO NOT CHANGE
//  Depth & Number Of Address Bits For TX Buffer
//  ** NOTE ** : VUSB_HS_TX_DEPTH and VUSB_HS_TX_ADD must be updated when either
//               VUSB_HS_TX_CHAN or VUSB_HS_DEV_EP are changed.
//  Host Only : VUSB_HS_TX_DEPTH = VUSB_HS_TX_CHAN
//  Device    : VUSB_HS_TX_DEPTH = (VUSB_HS_TX_CHAN)*min(1,VUSB_DEV_EP)
//          : VUSB_HS_TX_DEPTH <--> ** Power of 2 not necessary ** ; integer multiple of power
of 2 acceptable;
//                                  ie. 2**X * VUSB_HS_DEV_EP.
//                             (verify code exists for chosen TX_DEPTH in graycode package file)
parameter VUSB_HS_TX_DEPTH = 128; //  (VUSB_HS_TX_CHAN)*max_int(1,VUSB_HS_DEV_EP);
parameter VUSB_HS_TX_ADD = 7; //  log2(VUSB_HS_TX_DEPTH);
parameter VUSB_HS_TX_BURST = 8; //  Burst size for Memory To TX Buffer Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
//
//  Device/OTG : This constant determines if the device endpoint
//  context data used for rapid transmit response (reduced latency) and
//  double buffering tracking is stored in the TX-FIFO or if it is held
//  in registers.  There are advantages and disadvantages to each configuration:
//
//  = 0 (context in FIFO)
//     advantage    : smaller gate count & power (less 126 registers + misc. per endpoint)
//    disadvantage : actual latency buffer size is smaller because 4 words are used for context
(ie. actual VUSB_HS_TX_CHAN-4 words)
//                   FIFO interface is 2 read ports / 1 write port***
//
//  = 1 (context is registers)
//     advantage    : latency buffer uses full VUSB_HS_TX_CHAN words for data
//                   FIFO interface is 1 read port / 1 write port***
//     disadvantage : larger gate count & power (adds 144 registers + misc.)
//
//  *** this difference MAY be significant with some technologies and MAY offset the gate count
and power difference.
//
parameter VUSB_HS_TX_LOCAL_CONTEXT_REGISTERS = 1; //  [0 - context in FIFO, 1 - context in
registers]
```

```
// ------------------------------------------------------------------------
//  Device Version Number
// ------------------------------------------------------------------------
parameter VUSB_HS_DCIVERSION = 16'b 0000000000000001;
parameter VUSB_HS_HCIVERSION = 16'b 0000000100000000; //  EHCI 1.00
```

60.3.2.2 Host Port 2 Signal Connections and Signal Muxing

Table 60-12 details the I/O pad connections for the USB Host Port 2. The direction control is logic 1 for output.

Table 60-12. Host Port 2 Signal Connections

I/O PAD	Type	Input	Output	Direction control
USB2_ULPI_CLK	I/O	ipp_ind_uh2_clk	ipp_do_uh2_clk	ipp_obe_uh2_clk
USB2_ULPI_DIR	I/O	ipp_ind_uh2_dir	ipp_do_uh2_dir	ipp_obe_uh2_dir
USB2_ULPI_STP	I/O	ipp_ind_uh2_stp	ipp_do_uh2_stp	ipp_obe_uh2_stp
USB2_ULPI_NXT	I/O	ipp_ind_uh2_nxt	ipp_do_uh2_nxt	ipp_obe_uh2_nxt
USB2_ULPI_DATA0	I/O	ipp_ind_uh2_data0	ipp_do_uh2_data0	ipp_obe_uh2_data0
USB2_ULPI_DATA1	I/O	ipp_ind_uh2_data1	ipp_do_uh2_data1	ipp_obe_uh2_data1
USB2_ULPI_DATA2	I/O	ipp_ind_uh2_data2	ipp_do_uh2_data2	ipp_obe_uh2_data2
USB2_ULPI_DATA3	I/O	ipp_ind_uh2_data3	ipp_do_uh2_data3	ipp_obe_uh2_data3
USB2_ULPI_DATA4	I/O	ipp_ind_uh2_data4	ipp_do_uh2_data4	ipp_obe_uh2_data4
USB2_ULPI_DATA5	I/O	ipp_ind_uh2_data5	ipp_do_uh2_data5	ipp_obe_uh2_data5
USB2_ULPI_DATA6	I/O	ipp_ind_uh2_data6	ipp_do_uh2_data6	ipp_obe_uh2_data6
USB2_ULPI_DATA7	I/O	ipp_ind_uh2_data7	ipp_do_uh2_data7	ipp_obe_uh2_data7

Host port 2 can support either a ULPI transceiver or a serial transceiver. Depending on the selected type, signaling for one or the other is available at the top level. Table 60-15 shows the relation between the top-level signals and the USB core signals in both modes.

Table 60-13. / Serial Muxing

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	Serial Mode	ULPI Mode	To I/O Mux	Serial Mode	ULPI Mode
ipp_ind_uh2_data0	ipp_ind_uh2_oe_n	uh2_data_in[0]	ipp_do_uh2_data0	ipp_do_uh2_oe_n	uh2_data_out[0]
ipp_ind_uh2_data1	ipp_ind_uh2_rxvm_txdm_se0	uh2_data_in[1]	ipp_do_uh2_data1	ipp_do_uh2_txdp_dat	uh2_data_out[1]
ipp_ind_uh2_data2	ipp_ind_uh2_rxvp_txdp_dat	uh2_data_in[2]	ipp_do_uh2_data2	ipp_do_uh2_txdm_se0	uh2_data_out[2]
ipp_ind_uh2_data3	—	uh2_data_in[3]	ipp_do_uh2_data3	ipp_do_uh2_suspend	uh2_data_out[3]

Table 60-13. / Serial Muxing (continued)

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	Serial Mode	ULPI Mode	To I/O Mux	Serial Mode	ULPI Mode
ipp_ind_uh2_data4	ipp_ind_uh2_rxvp_txdp_dat	uh2_data_in[4]	ipp_do_uh2_data4	ipp_do_uh2_rxvp_txdp_dat	uh2_data_out[4]
ipp_ind_uh2_data5	ipp_ind_uh2_rxvm_txdm_se0	uh2_data_in[5]	ipp_do_uh2_data5	ipp_do_uh2_rxvm_txdm_se0	uh2_data_out[5]
ipp_ind_uh2_data6	ipp_ind_uh2_xcvr_ser_rcv	uh2_data_in[6]	ipp_do_uh2_data6	ipp_do_uh2_xcvr_ser_rcv	uh2_data_out[6]
ipp_ind_uh2_data7	—	uh2_data_in[7]	ipp_do_uh2_data7	ipp_do_uh2_speed	uh2_data_out[7]
			USB Port 2 Direction Control		
—	—	—	ipp_obe_uh2_data0	ipp_obe_uh2_oe_n	uh2_data_out_en
—	—	—	ipp_obe_uh2_data1	~uh2_oe_n	uh2_data_out_en
—	—	—	ipp_obe_uh2_data2	~uh2_oe_n	uh2_data_out_en
—	—	—	ipp_obe_uh2_data3	1’b0	uh2_data_out_en
—	—	—	ipp_obe_uh2_data4	ipp_obe_uh2_rxvp_txdp_dat	uh2_data_out_en
—	—	—	ipp_obe_uh2_data5	ipp_obe_uh2_rxvm_txdm_se0	uh2_data_out_en
—	—	—	ipp_obe_uh2_data6	ipp_obe_uh2_xcvr_ser_rcv	uh2_data_out_en
—	—	—	ipp_obe_uh2_data7	1'b0	uh2_data_out_en

60.3.3 USB Host Controller 3

The Host 3 core USB signals do not connect directly to the HOST3 I/O pins. Instead, the signals pass through the HS-ULPI TLL and serial and ULPI mux, which allows additional functionality on Host Port3. Host controller 3 is configured for operation with a FS/LS serial transceiver or a parallel ULPI transceiver (HS/FS/LS). The selection between transceiver type is software programmable. The module defaults to serial mode.

60.3.3.1 Host Controller 3 Configuration Parameter

Host Controller 3 is configured for high-speed/full-speed/low-speed operation and supports serial transceiver interface and ULPI interface. The parameters shown in Example 60-3 are used to set Host Controller 3 core implementation.

Example 60-3. Host Controller 3 Core Implementation

```
// ------------------------------------------------------------------
// ------------------------------------------------------------------------
//  System Configuration Options
// ------------------------------------------------------------------------
//
//  The VUSB_HS_RESET_TYPE constant determines the reset type used in the core.
//    0 = Use Synchronous Resets
//    1 = Use Asynchronous Resets
//
//  The VUSB_HS_RESET_OPTIONAL constant determines if various register files
//  are connected to the main reset.
//
//  0 = Reset all flops except those listed above.
//  1 = Reset 100% of flops.
//
parameter VUSB_HS_RESET_TYPE = 1; //  [0,1]
parameter VUSB_HS_RESET_OPTIONAL = 0; //  [0,1]
//
//  The VUSB_HS_CLOCK_CONFIGURATION constant determines the clocking used in the core
//  See reference manual for a description of the clocking modes available.
//
//    0 = xcvr_clk_0 = pe_clk = clk (do not select for multi-port host product)
//    1 = xcvr_clk_0 < pe_clk = clk
//    2 = xcvr_clk_0 = pe_clk <> clk (do not select for multi-port host product)
//    3 = xcvr_clk_0 < pe_clk <> clk
parameter VUSB_HS_CLOCK_CONFIGURATION = 2; //  [0,1,2,3]
//
//  phy size (UTMI Only)
//  Set according to the data width of the transciever connected to the core.
//    0 = 8 bit wide data bus [60MHz clock from the transciever]
//    1 = 16 bit wide data bus [30MHZ clock from the transciever]
//    2 = software programmable reset to 8-bit width
//    3 = software programmable reset to 16-bit width
//  NOTE: options 2,3 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
parameter VUSB_HS_PHY16_8 = 0; //  [0,1,2,3]
//  This constant selects which phy is being used
//  0 = UTMI/UMTI+
//  1 = Philips legacy style
//  2 = ULPI ** Must have purchased the ULPI Option **
//  3 = Serial Only
//  4 = Software programmable - reset to UTMI
//  5 = Software programmable - reset to Philips legacy style
//  6 = Software programmable - reset to ULPI ** Must have purchased the ULPI Option **
//  7 = Software programmable - reset to Serial
//  NOTE: options 4,5,6,7 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
//  NOTE: undefined results if ULPI is selected (#2,6) and the ULPI option was not purchased &
installed.
parameter VUSB_HS_PHY_TYPE = 7; //  [0,1,2,3,4,5,6,7]
```

```
//  derived constants ; must set to match VUSB_HS_PHY_TYPE above according the attached
instructions.
parameter VUSB_HS_PHY_UTMI = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=1,2,3       else 1
parameter VUSB_HS_PHY_PHIL = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,2,3       else 1
parameter VUSB_HS_PHY_ULPI = 1; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,1,3       else 1
//  This constant selects that the serial engine is used in place of the
//  parallel signalling for UTMI+,Philips, and ULPI.
//  0 - No Serial Engine      - Always use parallel signalling
//  1 - Serial Engine Present - Always use serial signalling for FS/LS
//  2 - Software programmable - Reset to use parallel signalling for FS/LS
//  3 - Software programmable - Reset to use serial signalling for FS/LS
//  NOTE: Must set to 1 if PHY_TYPE=1 or 3
//  NOTE: Must set to 1 or 3 is PHY_TYPE=6
//  NOTE: option 0 slims gate count by removing all logic necessary in the serial engine
parameter VUSB_HS_PHY_SERIAL = 2;
// ------------------------------------------------------------------------
//  Device Configuration
// ------------------------------------------------------------------------
//
//  The VUSB_HS_DEV_EP constant represents the total number of endpoints.
//
//  ** NOTE ** : VUSB_HS_DEV_EP should be set to 1 for host only implementations,
//  ** NOTE ** : VUSB_HS_DEV_EP_ADD must be updated when the VUSB_HS_DEV_EP is changed.
//
parameter VUSB_HS_DEV_EP = 1; //  [2,3,...,16]
//  set the VUSB_HS_DEV_EP_ADD equal to max(1, log2(VUSB_HS_DEV_EP))
parameter VUSB_HS_DEV_EP_ADD = 1;
// ------------------------------------------------------------------------
//  Host Configuration
// ------------------------------------------------------------------------
//  The VUSB_HS_NUM_PORT constant specifies the number of downstream ports
//  supported by the host controller.  This constant is only relevant for
//  for the multi-port host product.  Leave 1 for device, otg, and single port host
//  products.
//
//  Maximum: 8 downstream ports.
//
parameter VUSB_HS_NUM_PORT = 1; //  [1,2,...,8]
// ------------------------------------------------------------------------
//  Host Configuration - Internal Transaction Translator
// ------------------------------------------------------------------------
//  These constants define how many periodic contexts the internal TT can support.
//
//  Note1: These constants are only relevent for the multi-port host product.
//  Note2: The number of async. contexts should not be adjusted.
//
//  USB 2.0 spec requires 16 periodic contexts for a hub TT although this seems unncessarily
//  big for an embedded application where limits can be imposed on the downstream applications.
//  Changing the  number from 16 to 4 will significantly reduce gate count but be cautioned
//  that this will impose limits on the number of downstream periodic (ISO/Interrupt)
transactions.
//
//  The simplifying limit imposed when the number of periodic contexts is reduced to 4 is
//  that the application can send no more than 4 periodic (ISO/Interrupt) packets per
//  frame to the downstream Full and Low-Speed devices. Whereas, when the number
//  of periodic contexts are 16, its possible to pipeline packets accross the
//  microframes and send as many periodic packets per frame as can be scheduled (> 16).
```

```
//
parameter VUSB_HS_TT_ASYNC_CONTEXTS = 2; //  DO NOT CHANGE
parameter VUSB_HS_TT_ASYNC_CONTEXTS_ADDR = 1; //  DO NOT CHANGE ;
max(1,log2(VUSB_HS_TT_ASYNC_CONTEXTS))
parameter VUSB_HS_TT_PERIODIC_CONTEXTS = 16; //  [4,16 Only] ; Caution (see note above)
//  if this is changed to 4 then change the following constants:
//     VUSB_HS_TT_PERIODIC_CONTEXT_ADDR=2
//     VUSB_HS_TT_CONTEXTS_ADDR=2
parameter VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR = 4; //  max(1,log2(VUSB_HS_TT_PERIODIC_CONTEXTS))
parameter VUSB_HS_TT_CONTEXTS_ADDR = 4; //  max(VUSB_HS_TT_ASYNC_CONTEXTS_ADDR,
VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR)
// ------------------------------------------------------------------------
//  RX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For RX Buffer
//  ** NOTE ** : VUSB_HS_RX_ADD must be updated when VUSB_HS_RX_DEPTH is changed.
parameter VUSB_HS_RX_DEPTH = 128; //  Must be power of 2! (ie. 2**j)
//  Minimum : >= 8
//  Minimum : >= 3*VUSB_HS_RX_BURST/2
//  Maximum : <= 2048
//  Recommended : > 2*VUSB_HS_RX_BURST
parameter VUSB_HS_RX_ADD = 7; //  log2(VUSB_HS_RX_DEPTH)
parameter VUSB_HS_RX_BURST = 8; //  Burst size for RX Buffer To Memory Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
// ------------------------------------------------------------------------
//  TX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For TX Channel
//  The Channel constants control the size of the buffer associated with each TX
//  endpoint channel.  All of the channel buffers are combined into a single
//  TX Buffer using the TX Buffer constants.
//  ** NOTE ** : VUSB_HS_TX_CHAN_ADD must be updated when VUSB_HS_TX_CHAN is changed.
parameter VUSB_HS_TX_CHAN = 128; //  Must be power of 2 (ie. 2**k)
//  Minimum : >= 16
//  Minimum : >= 3*VUSB_HS_TX_BURST/2+4
//  Maximum : <= 128
//  Recommended : > 2*VUSB_HS_TX_BURST+4
parameter VUSB_HS_TX_CHAN_ADD = 7; //  log2(VUSB_HS_TX_CHAN);
parameter VUSB_HS_TX_CHAN_STATE = 4; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_STATE_ADD = 2; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_DATA = VUSB_HS_TX_CHAN - VUSB_HS_TX_CHAN_STATE; //  DO NOT CHANGE
//  Depth & Number Of Address Bits For TX Buffer
//  ** NOTE ** : VUSB_HS_TX_DEPTH and VUSB_HS_TX_ADD must be updated when either
//               VUSB_HS_TX_CHAN or VUSB_HS_DEV_EP are changed.
//  Host Only : VUSB_HS_TX_DEPTH = VUSB_HS_TX_CHAN
//  Device    : VUSB_HS_TX_DEPTH = (VUSB_HS_TX_CHAN)*min(1,VUSB_DEV_EP)
//          : VUSB_HS_TX_DEPTH <--> ** Power of 2 not necessary ** ; integer multiple of power
of 2 acceptable;
//                                    ie. 2**X * VUSB_HS_DEV_EP.
//                              (verify code exists for chosen TX_DEPTH in graycode package file)
parameter VUSB_HS_TX_DEPTH = 128; //  (VUSB_HS_TX_CHAN)*max_int(1,VUSB_HS_DEV_EP);
parameter VUSB_HS_TX_ADD = 7; //  log2(VUSB_HS_TX_DEPTH);
parameter VUSB_HS_TX_BURST = 8; //  Burst size for Memory To TX Buffer Transfers
//  Minimum : 2
//  Maximum : 128
```

```
//  Recommended : >= 4
//
//  Device/OTG : This constant determines if the device endpoint
//  context data used for rapid transmit response (reduced latency) and
//  double buffering tracking is stored in the TX-FIFO or if it is held
//  in registers.  There are advantages and disadvantages to each configuration:
//
//  = 0 (context in FIFO)
//     advantage    : smaller gate count & power (less 126 registers + misc. per endpoint)
//    disadvantage : actual latency buffer size is smaller because 4 words are used for context
(ie. actual VUSB_HS_TX_CHAN-4 words)
//                   FIFO interface is 2 read ports / 1 write port***
//
//  = 1 (context is registers)
//     advantage    : latency buffer uses full VUSB_HS_TX_CHAN words for data
//                   FIFO interface is 1 read port / 1 write port***
//     disadvantage : larger gate count & power (adds 144 registers + misc.)
//
//  *** this difference MAY be significant with some technologies and MAY offset the gate count
and power difference.
//
parameter VUSB_HS_TX_LOCAL_CONTEXT_REGISTERS = 1; //  [0 - context in FIFO, 1 - context in
registers]
// ------------------------------------------------------------------------
//  Device Version Number
// ------------------------------------------------------------------------
parameter VUSB_HS_DCIVERSION = 16'b 0000000000000001;
parameter VUSB_HS_HCIVERSION = 16'b 0000000100000000; //  EHCI 1.00
```

60.3.3.2 Host Port 3 Signal Connections and Signal Muxing

Table 60-12 details the I/O pad connections for the USB Host Port 3. The direction control is logic 1 for output.

Table 60-14. Host Port 2 Signal Connections

I/O PAD	Type	Input	Output	Direction control
USB3_ULPI_CLK	I/O	ipp_ind_uh3_clk	ipp_do_uh3_clk	ipp_obe_uh3_clk
USB3_ULPI_DIR	I/O	ipp_ind_uh3_dir	ipp_do_uh3_dir	ipp_obe_uh3_dir
USB3_ULPI_STP	I/O	ipp_ind_uh3_stp	ipp_do_uh3_stp	ipp_obe_uh3_stp
USB3_ULPI_NXT	I/O	ipp_ind_uh3_nxt	ipp_do_uh3_nxt	ipp_obe_uh3_nxt
USB3_ULPI_DATA0	I/O	ipp_ind_uh3_data0	ipp_do_uh3_data0	ipp_obe_uh3_data0
USB3_ULPI_DATA1	I/O	ipp_ind_uh3_data1	ipp_do_uh3_data1	ipp_obe_uh3_data1
USB3_ULPI_DATA2	I/O	ipp_ind_uh3_data2	ipp_do_uh3_data2	ipp_obe_uh3_data2
USB3_ULPI_DATA3	I/O	ipp_ind_uh3_data3	ipp_do_uh3_data3	ipp_obe_uh3_data3
USB3_ULPI_DATA4	I/O	ipp_ind_uh3_data4	ipp_do_uh3_data4	ipp_obe_uh3_data4
USB3_ULPI_DATA5	I/O	ipp_ind_uh3_data5	ipp_do_uh3_data5	ipp_obe_uh3_data5
USB3_ULPI_DATA6	I/O	ipp_ind_uh3_data6	ipp_do_uh3_data6	ipp_obe_uh3_data6
USB3_ULPI_DATA7	I/O	ipp_ind_uh3_data7	ipp_do_uh3_data7	ipp_obe_uh3_data7

Host port 3 can support either a ULPI transceiver or a Serial Transceiver. Depending on the selected type, signaling for one or the other is available at the top level. Table 60-15 shows the relation between the top-level signals and the USB core signals in both modes.

Table 60-15. / Serial Muxing

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	**Serial Mode**	**ULPI Mode**	**To I/O Mux**	**Serial Mode**	**ULPI Mode**
ipp_ind_uh3_data0	ipp_ind_uh3_oe_n	uh3_data_in[0]	ipp_do_uh3_data0	ipp_do_uh3_oe_n	uh3_data_out[0]
ipp_ind_uh3_data1	ipp_ind_uh3_rxvm_txdm_se0	uh3_data_in[1]	ipp_do_uh3_data1	ipp_do_uh3_txdp_dat	uh3_data_out[1]
ipp_ind_uh3_data2	ipp_ind_uh3_rxvp_txdp_dat	uh3_data_in[2]	ipp_do_uh3_data2	ipp_do_uh3_txdm_se0	uh3_data_out[2]
ipp_ind_uh3_data3	—	uh3_data_in[3]	ipp_do_uh3_data3	ipp_do_uh3_suspend	uh3_data_out[3]
ipp_ind_uh3_data4	ipp_ind_uh3_rxvp_txdp_dat	uh3_data_in[4]	ipp_do_uh3_data4	ipp_do_uh3_rxvp_txdp_dat	uh3_data_out[4]
ipp_ind_uh3_data5	ipp_ind_uh3_rxvm_txdm_se0	uh3_data_in[5]	ipp_do_uh3_data5	ipp_do_uh3_rxvm_txdm_se0	uh3_data_out[5]
ipp_ind_uh3_data6	ipp_ind_uh3_xcvr_ser_rcv	uh3_data_in[6]	ipp_do_uh3_data6	ipp_do_uh3_xcvr_ser_rcv	uh3_data_out[6]
ipp_ind_uh3_data7	—	uh3_data_in[7]	ipp_do_uh3_data7	ipp_do_uh3_speed	uh3_data_out[7]
			USB Port 2 Direction Control		
—	—	—	ipp_obe_uh3_data0	ipp_obe_uh3_oe_n	uh3_data_out_en
—	—	—	ipp_obe_uh3_data1	~uh3_oe_n	uh3_data_out_en
—	—	—	ipp_obe_uh3_data2	~uh3_oe_n	uh3_data_out_en
—	—	—	ipp_obe_uh3_data3	1’b0	uh3_data_out_en
—	—	—	ipp_obe_uh3_data4	ipp_obe_uh3_rxvp_txdp_dat	uh3_data_out_en
—	—	—	ipp_obe_uh3_data5	ipp_obe_uh3_rxvm_txdm_se0	uh3_data_out_en
—	—	—	ipp_obe_uh3_data6	ipp_obe_uh3_xcvr_ser_rcv	uh3_data_out_en
—	—	—	ipp_obe_uh3_data7	1'b0	uh3_data_out_en

60.3.4 USB OTG Controller

The OTG controller offers HS/FS/LS capabilities in Host mode and HS/FS in device mode.

60.3.4.1 OTG Controller Configuration Parameter

OTG Controller is configured for high-speed/full-speed/low-speed operation and supports serial transceiver interface and ULPI interface. The parameters shown in Example 60-4 show how to set the OTG Controller core implementation.

Example 60-4. OTG Controller Core Implementation

```
// ------------------------------------------------------------------------
//  System Configuration Options
// ------------------------------------------------------------------------
//
//  The VUSB_HS_RESET_TYPE constant determines the reset type used in the core.
//    0 = Use Synchronous Resets
//    1 = Use Asynchronous Resets
//
//  The VUSB_HS_RESET_OPTIONAL constant determines if various register files
//  are connected to the main reset.
//
//  0 = Reset all flops except those listed above.
//  1 = Reset 100% of flops.
//
parameter VUSB_HS_RESET_TYPE = 1; //  [0,1]
parameter VUSB_HS_RESET_OPTIONAL = 0; //  [0,1]
//
//  The VUSB_HS_CLOCK_CONFIGURATION constant determines the clocking used in the core
//  See reference manual for a description of the clocking modes available.
//
//    0 = xcvr_clk_0 = pe_clk = clk (do not select for multi-port host product)
//    1 = xcvr_clk_0 < pe_clk = clk
//    2 = xcvr_clk_0 = pe_clk <> clk (do not select for multi-port host product)
//    3 = xcvr_clk_0 < pe_clk <> clk
parameter VUSB_HS_CLOCK_CONFIGURATION = 2; //  [0,1,2,3]
//
//  phy size (UTMI Only)
//  Set according to the data width of the transciever connected to the core.
//    0 = 8 bit wide data bus [60MHz clock from the transciever]
//    1 = 16 bit wide data bus [30MHZ clock from the transciever]
//    2 = software programmable reset to 8-bit width
//    3 = software programmable reset to 16-bit width
//  NOTE: options 2,3 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
parameter VUSB_HS_PHY16_8 = 0; //  [0,1,2,3]
//  This constant selects which phy is being used
//  0 = UTMI/UMTI+
//  1 = Philips legacy style
//  2 = ULPI ** Must have purchased the ULPI Option **
//  3 = Serial Only
//  4 = Software programmable - reset to UTMI
//  5 = Software programmable - reset to Philips legacy style
//  6 = Software programmable - reset to ULPI ** Must have purchased the ULPI Option **
//  7 = Software programmable - reset to Serial
//  NOTE: options 4,5,6,7 are not recommended when VUSB_HS_CLOCK_CONFIGURATION = 0
```

```
//  NOTE: undefined results if ULPI is selected (#2,6) and the ULPI option was not purchased &
installed.
parameter VUSB_HS_PHY_TYPE = 7; //  [0,1,2,3,4,5,6,7]
//  derived constants ; must set to match VUSB_HS_PHY_TYPE above according the attached
instructions.
parameter VUSB_HS_PHY_UTMI = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=1,2,3       else 1
parameter VUSB_HS_PHY_PHIL = 0; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,2,3       else 1
parameter VUSB_HS_PHY_ULPI = 1; //  [0,1] ; set to 0 if VUSB_HS_PHY_TYPE=0,1,3       else 1
//  This constant selects that the serial engine is used in place of the
//  parallel signalling for UTMI+,Philips, and ULPI.
//  0 - No Serial Engine      - Always use parallel signalling
//  1 - Serial Engine Present - Always use serial signalling for FS/LS
//  2 - Software programmable - Reset to use parallel signalling for FS/LS
//  3 - Software programmable - Reset to use serial signalling for FS/LS
//  NOTE: Must set to 1 if PHY_TYPE=1 or 3
//  NOTE: Must set to 1 or 3 is PHY_TYPE=6
//  NOTE: option 0 slims gate count by removing all logic necessary in the serial engine
parameter VUSB_HS_PHY_SERIAL = 2;
// ------------------------------------------------------------------------
//  Device Configuration
// ------------------------------------------------------------------------
//
//  The VUSB_HS_DEV_EP constant represents the total number of endpoints.
//
//  ** NOTE ** : VUSB_HS_DEV_EP should be set to 1 for host only implementations,
//  ** NOTE ** : VUSB_HS_DEV_EP_ADD must be updated when the VUSB_HS_DEV_EP is changed.
//
parameter VUSB_HS_DEV_EP = 8; //  [2,3,...,16]
//  set the VUSB_HS_DEV_EP_ADD equal to max(1, log2(VUSB_HS_DEV_EP))
parameter VUSB_HS_DEV_EP_ADD = 3;
// ------------------------------------------------------------------------
//  Host Configuration
// ------------------------------------------------------------------------
//  The VUSB_HS_NUM_PORT constant specifies the number of downstream ports
//  supported by the host controller.  This constant is only relevant for
//  for the multi-port host product.  Leave 1 for device, otg, and single port host
//  products.
//
//  Maximum: 8 downstream ports.
//
parameter VUSB_HS_NUM_PORT = 1; //  [1,2,...,8]
// ------------------------------------------------------------------------
//  Host Configuration - Internal Transaction Translator
// ------------------------------------------------------------------------
//  These constants define how many periodic contexts the internal TT can support.
//
//  Note1: These constants are only relevent for the multi-port host product.
//  Note2: The number of async. contexts should not be adjusted.
//
//  USB 2.0 spec requires 16 periodic contexts for a hub TT although this seems unncessarily
//  big for an embedded application where limits can be imposed on the downstream applications.
//  Changing the  number from 16 to 4 will significantly reduce gate count but be cautioned
//  that this will impose limits on the number of downstream periodic (ISO/Interrupt)
transactions.
//
//  The simplifying limit imposed when the number of periodic contexts is reduced to 4 is
//  that the application can send no more than 4 periodic (ISO/Interrupt) packets per
```

```
//  frame to the downstream Full and Low-Speed devices. Whereas, when the number
//  of periodic contexts are 16, its possible to pipeline packets accross the
//  microframes and send as many periodic packets per frame as can be scheduled (> 16).
//
parameter VUSB_HS_TT_ASYNC_CONTEXTS = 2; //  DO NOT CHANGE
parameter VUSB_HS_TT_ASYNC_CONTEXTS_ADDR = 1; //  DO NOT CHANGE ;
max(1,log2(VUSB_HS_TT_ASYNC_CONTEXTS))
parameter VUSB_HS_TT_PERIODIC_CONTEXTS = 16; //  [4,16 Only] ; Caution (see note above)
//  if this is changed to 4 then change the following constants:
//     VUSB_HS_TT_PERIODIC_CONTEXT_ADDR=2
//     VUSB_HS_TT_CONTEXTS_ADDR=2
parameter VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR = 4; //  max(1,log2(VUSB_HS_TT_PERIODIC_CONTEXTS))
parameter VUSB_HS_TT_CONTEXTS_ADDR = 4; //  max(VUSB_HS_TT_ASYNC_CONTEXTS_ADDR,
VUSB_HS_TT_PERIODIC_CONTEXTS_ADDR)
// ------------------------------------------------------------------------
//  RX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For RX Buffer
//  ** NOTE ** : VUSB_HS_RX_ADD must be updated when VUSB_HS_RX_DEPTH is changed.
parameter VUSB_HS_RX_DEPTH = 123; //  Must be power of 2! (ie. 2**j)
//  Minimum : >= 8
//  Minimum : >= 3*VUSB_HS_RX_BURST/2
//  Maximum : <= 2048
//  Recommended : > 2*VUSB_HS_RX_BURST
parameter VUSB_HS_RX_ADD = 7; //  log2(VUSB_HS_RX_DEPTH)
parameter VUSB_HS_RX_BURST = 8; //  Burst size for RX Buffer To Memory Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
// ------------------------------------------------------------------------
//  TX Buffer Constants
// ------------------------------------------------------------------------
//  Depth & Number Of Address Bits For TX Channel
//  The Channel constants control the size of the buffer associated with each TX
//  endpoint channel.  All of the channel buffers are combined into a single
//  TX Buffer using the TX Buffer constants.
//  ** NOTE ** : VUSB_HS_TX_CHAN_ADD must be updated when VUSB_HS_TX_CHAN is changed.
parameter VUSB_HS_TX_CHAN = 128; //  Must be power of 2 (ie. 2**k)
//  Minimum : >= 16
//  Minimum : >= 3*VUSB_HS_TX_BURST/2+4
//  Maximum : <= 128
//  Recommended : > 2*VUSB_HS_TX_BURST+4
parameter VUSB_HS_TX_CHAN_ADD = 7; //  log2(VUSB_HS_TX_CHAN);
parameter VUSB_HS_TX_CHAN_STATE = 4; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_STATE_ADD = 2; //  DO NOT CHANGE
parameter VUSB_HS_TX_CHAN_DATA = VUSB_HS_TX_CHAN - VUSB_HS_TX_CHAN_STATE; //  DO NOT CHANGE
//  Depth & Number Of Address Bits For TX Buffer
//  ** NOTE ** : VUSB_HS_TX_DEPTH and VUSB_HS_TX_ADD must be updated when either
//               VUSB_HS_TX_CHAN or VUSB_HS_DEV_EP are changed.
//  Host Only : VUSB_HS_TX_DEPTH = VUSB_HS_TX_CHAN
//  Device    : VUSB_HS_TX_DEPTH = (VUSB_HS_TX_CHAN)*min(1,VUSB_DEV_EP)
//          : VUSB_HS_TX_DEPTH <--> ** Power of 2 not necessary ** ; integer multiple of power
of 2 acceptable;
//                                    ie. 2**X * VUSB_HS_DEV_EP.
//                              (verify code exists for chosen TX_DEPTH in graycode package file)
parameter VUSB_HS_TX_DEPTH = 1024; //  (VUSB_HS_TX_CHAN)*max_int(1,VUSB_HS_DEV_EP);
parameter VUSB_HS_TX_ADD = 10; //  log2(VUSB_HS_TX_DEPTH);
```

```
parameter VUSB_HS_TX_BURST = 8; //  Burst size for Memory To TX Buffer Transfers
//  Minimum : 2
//  Maximum : 128
//  Recommended : >= 4
//
//  Device/OTG : This constant determines if the device endpoint
//  context data used for rapid transmit response (reduced latency) and
//  double buffering tracking is stored in the TX-FIFO or if it is held
//  in registers.  There are advantages and disadvantages to each configuration:
//
//  = 0 (context in FIFO)
//     advantage    : smaller gate count & power (less 126 registers + misc. per endpoint)
//    disadvantage : actual latency buffer size is smaller because 4 words are used for context
(ie. actual VUSB_HS_TX_CHAN-4 words)
//                   FIFO interface is 2 read ports / 1 write port***
//
//  = 1 (context is registers)
//     advantage    : latency buffer uses full VUSB_HS_TX_CHAN words for data
//                   FIFO interface is 1 read port / 1 write port***
//     disadvantage : larger gate count & power (adds 144 registers + misc.)
//
//  *** this difference MAY be significant with some technologies and MAY offset the gate count
and power difference.
//
parameter VUSB_HS_TX_LOCAL_CONTEXT_REGISTERS = 1; //  [0 - context in FIFO, 1 - context in
registers]
// ------------------------------------------------------------------------
//  Device Version Number
// ------------------------------------------------------------------------
parameter VUSB_HS_DCIVERSION = 16'b 0000000000000001;
parameter VUSB_HS_HCIVERSION = 16'b 0000000100000000; //  EHCI 1.00
```

60.3.4.2 Host Mode

The controller supports direct connection of a FS/LS device (without external hub) with external serial transceiver and HS/FS device with external ULPI transceiver. Although there is no separate Transaction Translator block in the system. The transaction translator function normally associated with a USB 2.0 high speed hub has been implemented within the DMA and Protocol engine blocks to support connection to full and low speed devices.

60.3.4.3 Peripheral (Device) Mode Features

The peripheral device mode features are as follows:

- Up to 8 bidirectional endpoints
- High/Full speed operation
- Supports HNP, SRP.
- Remote wakeup capable

60.3.4.4 Special Considerations

The OTG port functions as gateway between the Host 1 Port and the OTG transceiver when in Bypass mode.

60.3.4.5 OTG Port Signal Connections and Signal Muxing

Table 60-16 describes the Signal connections from the USB core to the OTG port OUTPUT PADS.

Table 60-16. Connections

I/O PAD	Type	Input	Output	Direction Control
USB_ULPI_CLK	I/O	ipp_ind_otg_clk	ipp_do_otg_clk	ipp_obe_otg_clk
USB_ULPI_DIR	IN	ipp_ind_otg_dir	—	—
USB_ULPI_STP	OUT	—	ipp_do_otg_stp	—
USB_ULPI_NXT	IN	ipp_ind_otg_nxt	—	—
USB_ULPI_DATA0	I/O	ipp_ind_otg_data0	ipp_do_otg_data0	ipp_obe_otg_data0
USB_ULPI_DATA1	I/O	ipp_ind_otg_data1	ipp_do_otg_data1	ipp_obe_otg_data1
USB_ULPI_DATA2	I/O	ipp_ind_otg_data2	ipp_do_otg_data2	ipp_obe_otg_data2
USB_ULPI_DATA3	I/O	ipp_ind_otg_data3	ipp_do_otg_data3	ipp_obe_otg_data3
USB_ULPI_DATA4	I/O	ipp_ind_otg_data4	ipp_do_otg_data4	ipp_obe_otg_data4
USB_ULPI_DATA5	I/O	ipp_ind_otg_data5	ipp_do_otg_data5	ipp_obe_otg_data5
USB_ULPI_DATA6	I/O	ipp_ind_otg_data6	ipp_do_otg_data6	ipp_obe_otg_data6
USB_ULPI_DATA7	I/O	ipp_ind_otg_data7	ipp_do_otg_data7	ipp_obe_otg_data7

Table 60-17. OTG ULPI/Serial Muxing

USB TOP Input signals			USB TOP Output Signals		
From I/O Mux	Serial Mode	ULPI Mode	To I/O Mux	Serial Mode	ULPI Mode
ipp_ind_otg_data0		otg_data_in[0]	ipp_do_otg_data0	ipp_do_otg_oe_n	otg_data_out[0]
ipp_ind_otg_data1	ipp_ind_otg_rxvm_txdm_se0	otg_data_in[1]	ipp_do_otg_data1	ipp_do_otg_txdp_dat	otg_data_out[1]
ipp_ind_otg_data2	ipp_ind_otg_rxvp_txdp_dat	otg_data_in[2]	ipp_do_otg_data2	ipp_do_otg_txdm_se0	otg_data_out[2]
ipp_ind_otg_data3	—	otg_data_in[3]	ipp_do_otg_data3	ipp_do_otg_suspend	otg_data_out[3]
ipp_ind_otg_data4	ipp_ind_otg_xcvr_ser_vp	otg_data_in[4]	ipp_do_otg_data4	ipp_do_otg_xcvr_ser_vp	otg_data_out[4]
ipp_ind_otg_data5	ipp_ind_otg_xcvr_ser_vm	otg_data_in[5]	ipp_do_otg_data5	ipp_do_otg_xcvr_ser_vm	otg_data_out[5]
ipp_ind_otg_data6	ipp_ind_otg_xcvr_ser_rcv	otg_data_in[6]	ipp_do_otg_data6	—	otg_data_out[6]
ipp_ind_otg_data7	—	otg_data_in[7]	ipp_do_otg_data7	ipp_do_otg_speed	otg_data_out[7]
—	—	—	OTG Port Direction Control		

Table 60-17. OTG ULPI/Serial Muxing (continued)

—	—	—	ipp_obe_otg_data0	1’b1	otg_data_out_en
—	—	—	ipp_obe_otg_data1	~otg_oe_n	otg_data_out_en
—	—	—	ipp_obe_otg_data2	~otg_oe_n	otg_data_out_en
—	—	—	ipp_obe_otg_data3	1’b0	otg_data_out_en
—	—	—	ipp_obe_otg_data4	ipp_obe_otg_xcvr_ser_vp	otg_data_out_en
—	—	—	ipp_obe_otg_data5	ipp_obe_otg_xcvr_ser_vm	otg_data_out_en
—	—	—	ipp_obe_otg_data6	1'b0	otg_data_out_en
—	—	—	ipp_obe_otg_data7	1'b0	otg_data_out_en

60.3.5 USB Power Control Module

The HS-USB module supports suspend and wakeup functionality, but the circuit is considered application specific and therefore not part of the IP. An external circuit has been designed to place external transceivers in suspend mode, and wake them up either on a local request (CPU initiated) or on remote request by detecting activity on the USB line. The wake-up logic can optionally wake the CPU when it is in sleep mode at the time of the request. The power control mechanism is described in the VUSB-HS-SPH and VUSB-HS-OTG reference manuals. For details on the power control module, please refer to the Power Module creation guide.

60.3.5.1 Entering Suspend Mode

Suspend mode is always entered under the control of driver software by setting the appropriate bit in PORTSC register. Once the controller is suspended, the clocks to the USB block can be stopped.

60.3.5.2 Wake-up Events

The power control module monitors the USB bus when the USB core is in the suspend state. Depending on whether the core is in host or device mode, a number of wakeup conditions are detected. Upon detection of a wakeup condition, an interrupt (asynchronous) is generated on the CPU complex. This interrupt also re-activates the clocks if these were stopped during the suspend.

60.3.5.2.1 Host Mode Events

The host controller wakes-up on the following events:

- Remote wakeup request—A peripheral can request the host to re-activate the bus by driving wake-up signaling on the Dm/Dp lines. The power control module detects a J-K transition on the Dm/Dp lines and signals the wakeup request to the core.
- Wake on overcurrent—If wake on overcurrent is enabled in the PORTSC registers, the power control module signals a wakeup condition to the USB core.

- Wake on disconnect—The power control module detects disconnect events by monitoring the Dp/Dm lines. When a disconnect event is detected (Dm = Dp = 0) and the wake on disconnect is enabled in the PORTSC register, the core is notified.
- Wake on Connect—Similar to wake on disconnect, the power control module detects a connect event (Dm or Dp High) and signals this to the USB core by setting the pwrctl_wakeup signal if enabled in the PORTSC register.

60.3.5.2.2 Device Mode Events

When the OTG controller is configured for peripheral operation, the power control module detects bus activity. Any non-idle condition on the USB bus activates the wakeup output of the power control module to notify the USB core of the wakeup event.

60.3.6 Transceiverless Link Logic (HS/FS-TLL) Module

The transceiverless link logic circuit allows two microcontrollers to use USB for an interprocessor communication link (ICL) without using conventional USB transceivers. The TLL muxes support ULPI (HS-TLL) and serial (FS-TLL) type interfacing and are available on Host Port 1 and Host Port 2.

60.3.6.1 TLL Mode Features

The features of this TLL mode include the following:

- HS/FS-TLL. HS-TLL and FS-TLL are selectable based on the type of USB peripheral. HS-TLL support HS-ULPI interface protocol, and FS-TLL support FS/LS serial interface protocol.
- TLL logic can be bypassed. When the bypass bit in the USB CONTROL register is enabled, the TLL module is bypassed and host USB core works in conventional USB mode.
- TLL meets the timing requirements of both host and device.

60.3.6.2 Serial FS-TLL Functional Description

The Serial FS-TLL logic is a logic representation of two serial transceivers connected by a USB cable. The USB bus DM/DP states are modeled internally in the FS-TLL function, such that the USB I/O port acts as if it were a transceiver.

In a regular USB implementation with serial PHYs, the speed selection on the USB bus is done by means of a pull-up resistor on the peripheral side either on the DM (low speed) or the DP (full Speed) line. This pull-up pulls one of the USB lines high when the bus is idle.

This option cannot be modeled in logic as the serial USB interface does not provide for such a signal. The FS-TLL mux is therefore configured for Full Speed only operation.

The IDLE condition of the bus is determined by the OEn signals and suspend signals on both sides of the mux. When both OEn signals are high, or when one or both suspend signals are high, the idle condition is assumed. The FS-TLL block then drives TxDp high and TxDm low on both sides of the block.

Figure 60-9 shows the FS-TLL mux functional diagram.

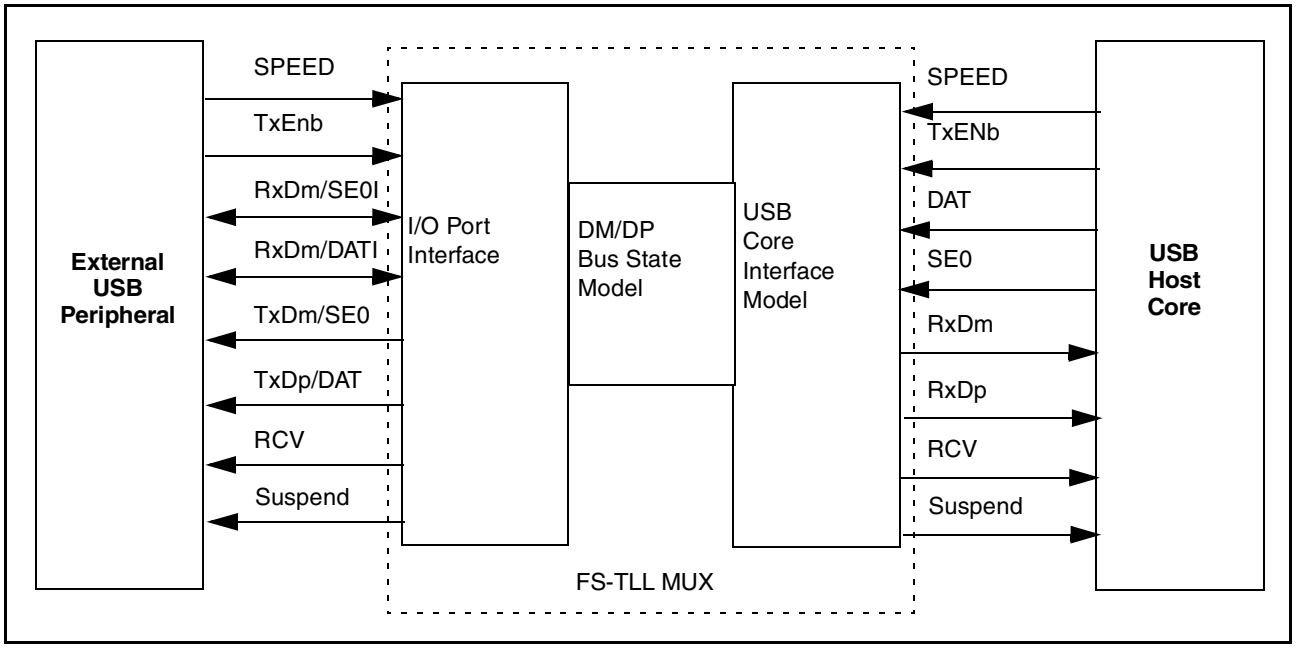


Figure 60-9. FS-TLL Mux Functional Diagram

60.3.6.3 ULPI HS-TLL Functional Description

ULPI HS-TLL module is an additional logic for the Host USB core with ULPI interface. It allows a HS-USB host communication with on-board HS-USB device (on-platform processors and modems) directly without ULPI PHY. Figure 60-10 show the functional diagram of ULPI HS-TLL.

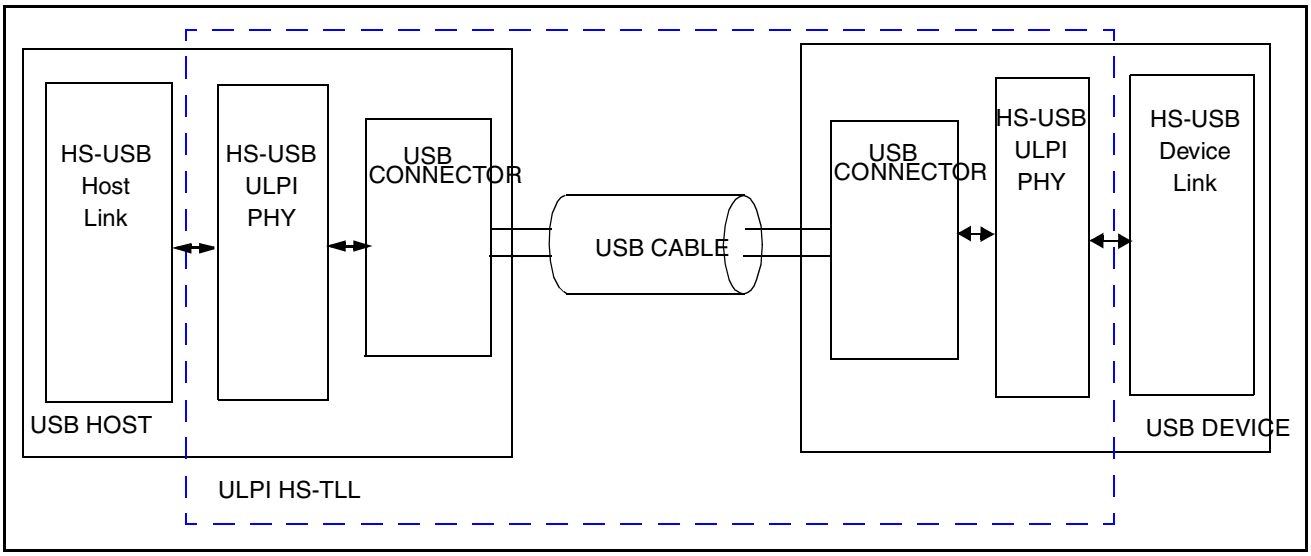


Figure 60-10. HS-TLL Functional Diagram

Figure 60-11 shows the block diagram combining ULPI interface and HS_TLL.

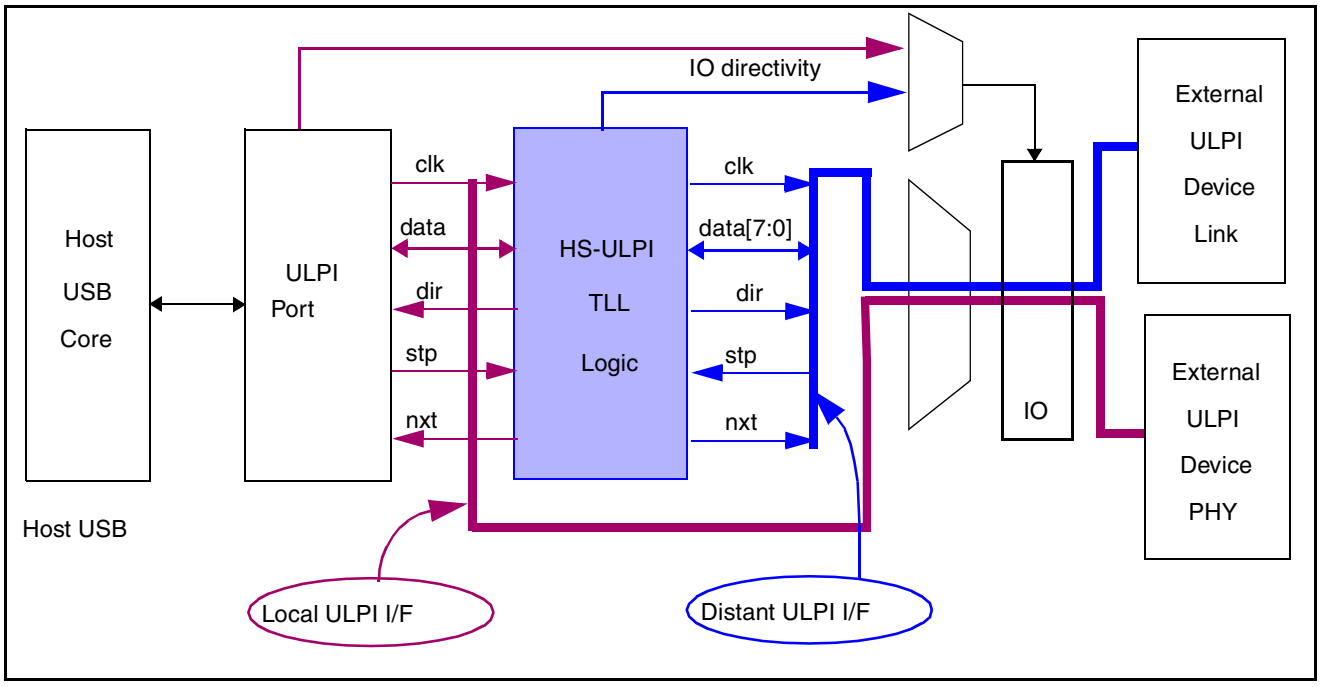


Figure 60-11. Block Diagram Combining ULPI Interface and HS-TLL

ULPI HS-TLL is designed to complied with the ULPI specification V1.1. Figure 60-12 shows a detailed block diagram of this module in USBOH3.

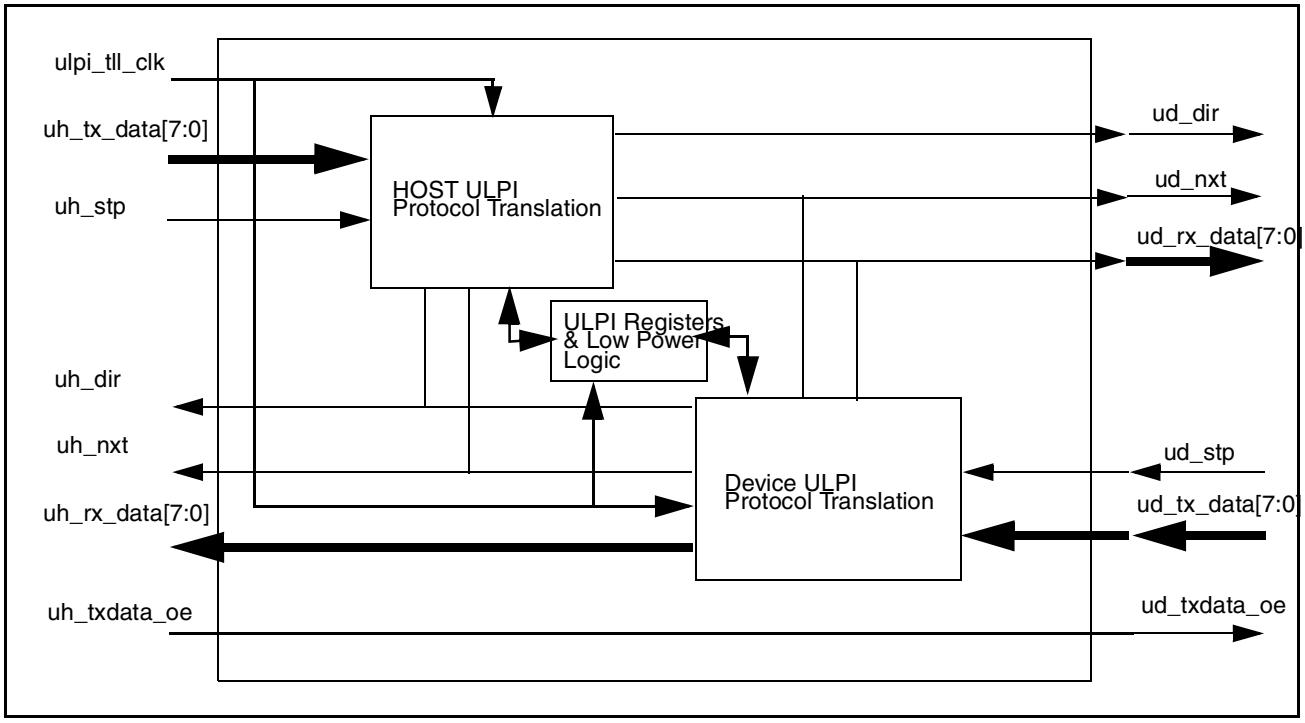


Figure 60-12. ULPI HS-TLL Detail Block Diagram

60.3.6.4 HS-TLL Detailed Features Definition

This section discusses the following:

- Section 60.3.6.4.1, General”
- Section 60.3.6.4.2, HS-TLL Data Transfer Protocol”
- Section 60.3.6.4.3, HS-TLL Modes Function”
- Section 60.3.6.4.4, Timing Requirements”
- Section 60.3.6.4.5, HS-TLL Programming Model”

60.3.6.4.1 General

In general HS-TLL imitates a host-side ULPI I/F to the local link layer, and a device-side ULPI I/F towards a link layer of a distant device. Unless otherwise said in the following sections, these two ports of the HS-TLL meet the full requirements of the ULPI specification[9].

However, several modes and features of the ULPI are not required, as discussed in the following sections. These modes and features are removed for the following reasons:

- On-board USB applications is the only use case of the TLL. No hot-insertion, plug-and-play scenarios are expected.
- There is no real analog PHY in between.

- Trial to keep HS-TLL definition compact.
- No OTG scenarios. Local processor is always a host, and distant processor is always a peripheral.

60.3.6.4.2 HS-TLL Data Transfer Protocol

HS-TLL supports ULPI protocol toward both local and distant links. This includes bus ownership, data transfer and abort protocols, as described by section 2.3 of ULPI Specification V1.1[9]. Within HS-TLL protocols of both sides are tied together such that data-transfer protocols are simultaneously operating to create a flowing link-to-link data transfer. Similarly, data abortion initiated by one of the links is translated by the HS-TLL to a PHY-like initiated data abortion in the other ULPI side.

Figure 60-13 shows the HS-USB packets data transmit from Host Link to Device Link through HS-TLL module. The packets data transmit from Device Link to Host Link through HS-TLL is the same.

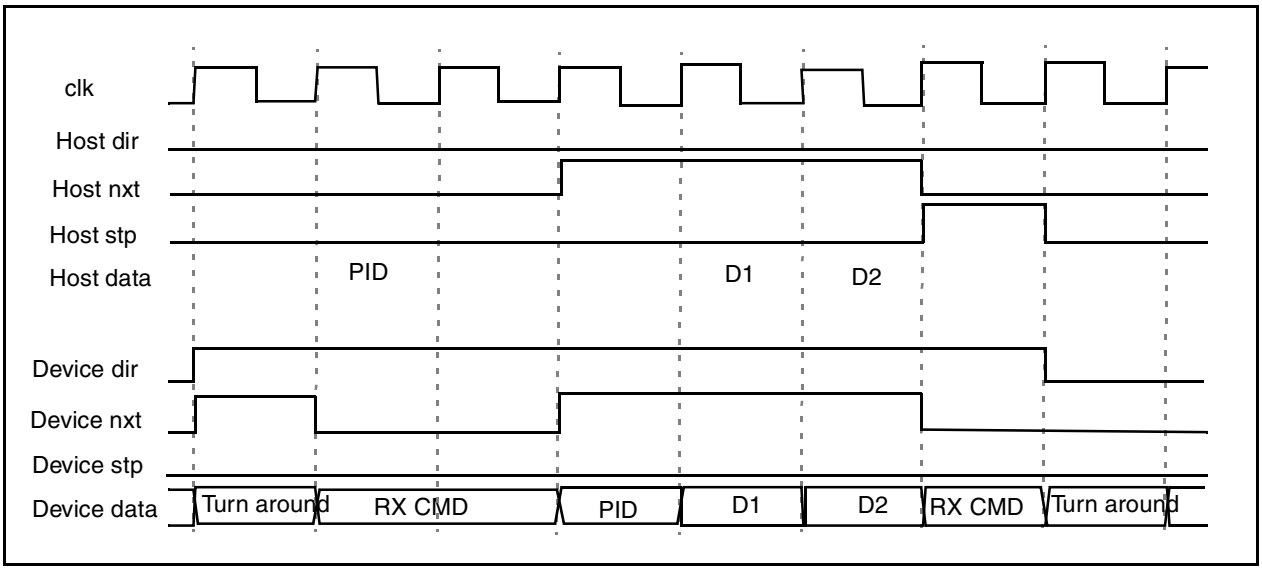


Figure 60-13. Data transmit(PID) from Host to Device through HS-TLL waveform

60.3.6.4.3 HS-TLL Modes Function

ULPI TLL supports only the basic synchronous mode and low-power mode. The trials of link side to change HS-TLL to one of the unsupported modes are ignored and generate a non-supported exception in the HS-TLL status register.

Preamble mode is not supported in the HS-TLL. The HS-TLL only operates in high speed ULPI mode; it does not support full-speed and low-speed mode. Only 8-bit mode within synchronous mode is supported at the local side and distant side ULPI I/F in this HS-TLL implementation. “6-pin”, “3-pin,” and “carkit” modes are not supported by current design, as they can by achieved by the non-TLL ULPI and FS-TLL functions of the USBOH3.

This implementation does support the following:

- High-speed chirp-like activities, as described in chapter 3.8.5 of ULPI Specification V1.1[9]
- High-Speed Remote wake up

- Resume

PID and non PID data streams from the USBHO2 core path through HS-TLL and are reflected to an external peripheral link according to the description in section 3.8.2 of ULPI Specification V1.1[9].

60.3.6.4.4 Timing Requirements

HS-TLL meets PHY-side timing requirements toward ULPI I/Fs of both local a distant links, as it is specified in Section 3.7 of ULPI Specification V1.1[9].

60.3.6.4.5 HS-TLL Programming Model

HS-TLL module supports function control and interface control registers access for ULPI control and status, but does not support other registers defined in ULPI description. During write, the unsupported register are ignored and during read, these register return the default value.

The ULPI standard defines a register set for the PHY side of the ULPI connection. The following subsections describe this part of the PHY register as implemented and reflect meaningful values of both the local and distant link layers:

Function Control Register (0x4–0x6)

Implementation is as follows:

- Bit 5—Reset

  This bit, and the related ULPI signals and protocol, is fully implemented for both distant and local links.
- Bit 6—SuspendM

  This bit, and the related ULPI signals and protocol, is fully implemented for both distant and local links. A careful design definition combines SuspendM from local link, SuspendM from distant link, and the HS-TLL own enable bit to define HS-TLL logic activity.

Interface Control Register (0x7–0x9):

Implementation is as follows:

- bit 4—AutoResume

  This bit, and the related ULPI signals and protocol, is not implemented for both distant and local links.
- USB Interrupt Enable Rising/Enable Falling/Status/Latch (0xD–0x14), bit 2,3—SessEnd, SessValid

  The 2 bits of each of these registers are supported to emulate session status as in a real PHY device for both distant and local links.
- Transmit and Receive Command bytes

  HS-TLL supports all commands to write and read the registers. For some unsupported registers, the write operation is ignored and the read operation returns the default value.

Tx CMD

Implementation is as follows:

- Reception of TX CMDs from local & distant ULPI I/Fs are supported:
- Special, Transmit (including PID codes), RegWrite, and RegRead command types are decoded and respond as defined in ULPI Specification V1.1[9].

Rx CMD

Implementation is as follows:

- Sending of RX CMDs to local and distant ULPI I/Fs are supported:
- LineState and RxEvent fields of the command are active and reflect relevant values according to data stream and protocol.
- VbusState, alt_int, and ID fields of the command are not supported and constantly reflect their inactive values.

Clock

Implementation is as follows:

- HS-TLL is always set to receive clk for local links (“Input Clock Mode” in 3.7.1.2 of ULPI Specification V1.1) and send clk for distant links (“Output Clock Mode”).
- Output Clock Mode at local ULPI and “Input Clock Mode” at distant ULPI are not supported. This is the CLK signal directivity also described at Figure 60-11.

Low-Power Modes

HS-TLL supports low-power mode entering and exiting, with local and distant link devices, independently. Low-power status change in one side of the TLL is reflected as linestate[1:0] and int changes in the other side of the TLL.

When in low-power mode, DATA[3:0] signals reflects the relevant state towards the link.

There are two ways to exit low-power mode when USB system is in suspend mode. When the host side is suspended and enters low-power mode, it can set the register bit in USB Control to cause the host to exit low-power mode and then resume the device. The peripheral side can also do remote wakeup by setting the PORTSC1 register soft wakeup bit to let the system exit suspend mode.

HS-TLL USB Operations

Figure 60-14 shows the sequence of HS-TLL USB Operations.

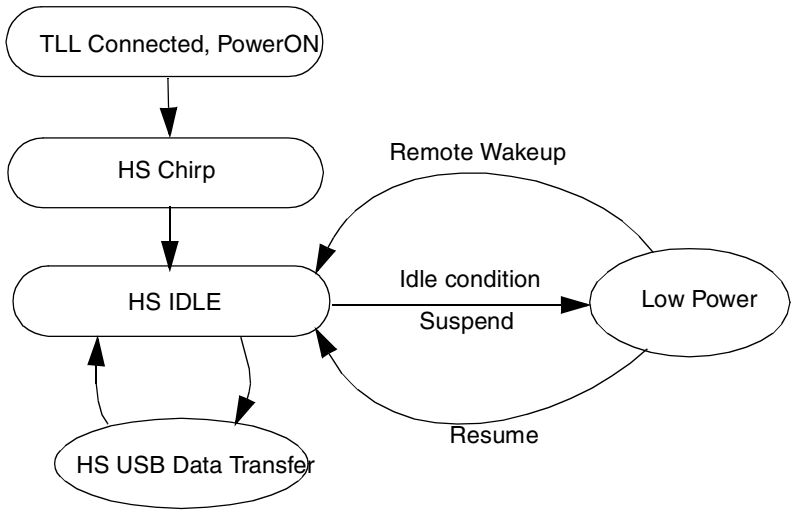


Figure 60-14. HS-TLL USB Operations

60.3.6.5 Host Port 1

On Host Port1, the FS-TLL function is integrated with the bypass function (see Section 60.3.7, USB Bypass Mode") and the transceiver type conversion logic. FS-TLL mode is disabled for this port as default. It can be enabled by setting bit 4 in the USBCONTROL register.

Table 60-18 shows the pin connections for FS-TLL mode and normal mode.

Table 60-18. Port 1 FS-TLL and PHY Mode Pin Connections

	FS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
SPEED	I	speed	O	speed
TxEnb	I	TxEnb	O	TxEnb
RxVm	I	TxDm	I	RxVm
RxVp	I	TxDp	I	RxVp
TxDm	O	RxDm	O	TxDm
TxDp	O	RxDp	O	TxDp
RCV	O	RCV	I	RCV
Suspend	I	suspend	O	SuspendM

The HS-TLL is only used in ULPI PHY. When in ULPI mode and TLL enable, the TLL function is enabled. Table 60-19 shows the HS-TLL pin connections.

Table 60-19. HS-TLL Pin Connection

	HS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
DIR	O	DIR	I	DIR
NXT	O	NXT	I	NXT
STP	I	STP	O	STP
DATA[7:0]	I/O	DATA[7:0]	I/O	DATA[7:0]

60.3.6.6 Host Port 2

The FS-TLL module on Host Port 2 contains the FS-TLL logic and the Serial Transceiver Type conversion logic. The interface type conversion is available in both TLL and Non TLL modes.

FS-TLL operation is disabled as the default mode. It can be turned on for operation with an external transceiver by setting bit 1 in th USB_UH2_CTRL register.

Table 60-20 shows the Port 2 FS-TLL and PHY mode pin connections.

Table 60-20. Port 2 FS-TLL and PHY Mode Pin Connections

	FS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
SPEED	I	speed	O	speed
TxEnb	I	TxEnb	O	TxEnb
RxVm	I	TxDm	I	RxVm
RxVp/	I	TxDp	O	RxVp
TxDm	O	RxDm	O	TxDm
TxDp	O	RxDp	O	TxDp
RCV	O	RCV	I	RCV
Suspend	I	suspend	O	SuspendM

The HS-TLL is only used in ULPI PHY, is the same as Host1. When in ULPI mode and TLL enable, the TLL function is enabled. Table 60-21 shows the HS-TLL pin connection.

Table 60-21. HS-TLL Pin Connection

	HS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
DIR	O	DIR	I	DIR
NXT	O	NXT	I	NXT
STP	I	STP	O	STP
DATA[7:0]	I/O	DATA[7:0]	I/O	DATA[7:0]

60.3.6.7 Host Port 3

The FS-TLL module on Host Port 3 contains the FS-TLL logic and the Serial Transceiver Type conversion logic. The interface type conversion is available in both TLL and Non TLL modes.

FS-TLL operation is disabled as the default mode. It can be turned on for operation with an external transceiver by setting bit 1 in th USB_UH3_CTRL register.

Table 60-22 shows the Port 3 FS-TLL and PHY mode pin connections.

Table 60-22. Port 3 FS-TLL and PHY Mode Pin Connections

	FS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
SPEED	I	speed	O	speed
TxEnb	I	TxEnb	O	TxEnb
RxVm	I	TxDm	I	RxVm
RxVp/	I	TxDp	O	RxVp
TxDm	O	RxDm	O	TxDm
TxDp	O	RxDp	O	TxDp
RCV	O	RCV	I	RCV
Suspend	I	suspend	O	SuspendM

The HS-TLL is only used in ULPI PHY, which is the same as Host1. When in ULPI mode and TLL enable, the TLL function is enabled.

Table 60-23 shows the HS-TLL mode pin connections.

Table 60-23. HS-TLL Pin Connection

	HS-TLL mode		PHY Mode	
Pin	**I/O**	**External Device Pin**	**I/O**	**External PHY Pin**
DIR	O	DIR	I	DIR
NXT	O	NXT	I	NXT
STP	I	STP	O	STP
DATA[7:0]	I/O	DATA[7:0]	I/O	DATA[7:0]

60.3.7 USB Bypass Mode

The USB Bypass mode is a special mode that allows for the transceiver on the OTG port to be used as transceiver for a USB peripheral device connected to host port 1. This mode is available for full-/low-speed serial and high-speed ULPI transceivers.

The bypass module is a combination of the following:

- The bypass function (Host Port1—OTG port pass through)
- FS-TLL function for Host Port 1
- Full-/low-bypass route with serial transceiver
- High-speed bypass route with ULPI transceiver
- Serial Transceiver Interface Conversion

60.3.7.1 Bypass Mode Operation

Bypass mode is enabled by writing a 1 to bit 0, as shown in Figure 60-2. Bypass mode supports serial or ULPI mode according to on-board peripheral and OTG external transceiver type.

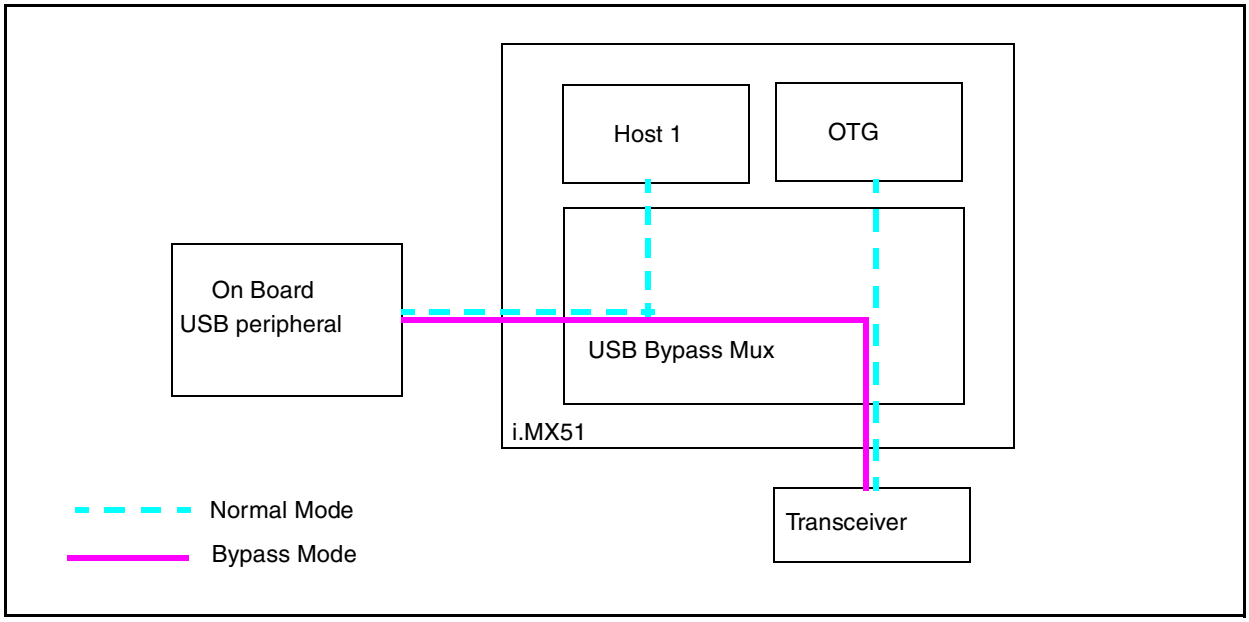


Figure 60-15. USB Bypass Mux Functional Diagram

In bypass mode, the interface signals (Serial/ULPI) from Host Port 1 are routed to the interface pins Serial/ULPI) of the OTG port such that an external USB Peripheral device can use the OTG transceiver (Serial/ULPI) to connect to an external USB Host. As this function only works with USB peripherals directly connected to Host Port 1, the port is automatically set for TLL mode. Transceiver Interface Type conversion is available in Bypass mode such that the interface type of the OTG transceiver can be different from the Host 1 interface type.

60.3.7.1.1 Serial Bypass mode

The USB HOST1 core is disconnected from the port and the inputs RxDp, RCV and RxDm from the core are driven by bits 9 and 10 (H1BPVAL) in Figure 60-2. The OTG core is also disconnected from its port and inputs RxDp, RCV, and RxDm are driven by bits 25 and 26 (OBPVAL) from Figure 60-2.

Transceiver Interface Type conversion is available in Bypass mode such that the interface type of the OTG transceiver can be different from the Host 1 interface type.

Table 60-24 list the pin functions of the Host 1 port when serial bypass mode is enabled and the associated OTG pin. The pin functions of the OTG port are not affected by serial bypass mode.

Table 60-24. HOST1/OTG Serial Bypass Mode Pin functions

	Unidirectional				Bidirectional				OTG PORT	
Pin	I/O	Single-Ended	I/O	Differential	I/O	Single- Ended	I/O	Differential	I/O	Pin
RxDm	I	RxDm	I	RxDm	I/O	SEOI/SE0O	I/O	RxDm/TxDm	O	TxDm
RxDp	I	RxDp	I	RxDp	I/O	DATI/DATO	I/O	RxDp/TxDP	O	TxDP
RCV	O	RCV	O	RCV		-	O	RCV	I	RCV
TxDm	O	SE0	O	TxDM		-		-	I	RxDM
TxDp	O	DAT	O	TxDP		-		-		RxDp
OEb	I	OEn	I	OEb	I	OEn	I	OEn		OEb
FS	I	Speed	I	FS	I	Speed	I	Speed		Speed
Suspend	I	Suspend	I	Suspend	I	Suspend	I	Suspend		Suspend

60.3.7.1.2 ULPI Bypass mode

The Host1 core is disconnected from the port and the OTG core is also disconnected from its port, then the external on-board USB peripheral with ULPI interface can route to the OTG side ULPI PHY transceiver, and the both USB USB peripheral and ULPI PHY clock are allowed and can configure by the USB Control register. Table 60-25 shows the Host1 and OTG ULPI bypass pin functions.

Table 60-25. HOST1/OTG ULPI Bypass Mode Pin functions

Signal Name	Host1 HS-TLL side pin		OTG side pin	
Data[7:0]	I/O	ipp_ind_uh1_data_tll[7:0] ipp_do_uh1_data_tll[7:0] ipp_obe_uh1_data_tll[7:0]	I/O	ipp_ind_otg_data[7:0] ipp_do_otg_data[7:0] ipp_obe_otg_data[7:0]
clock	I/O	ipp_ind_uh1_clk_tll ipp_do_uh1_clk_tll ipp_obe_uh1_clk_tll	I/O	ipp_ind_otg_clk ipp_do_otg_clk ipp_obe_otg_clk
dir	O	ipp_do_uh1_dir	I	ipp_ind_otg_dir
nxt	O	ipp_do_uh1_nxt	I	ipp_ind_otg_nxt
stp	I	ipp_ind_uh1_stp	O	ipp_do_otg_stp

Figure 60-16 shows the ULPI bypass mode Host1 and OTG pins route diagram.

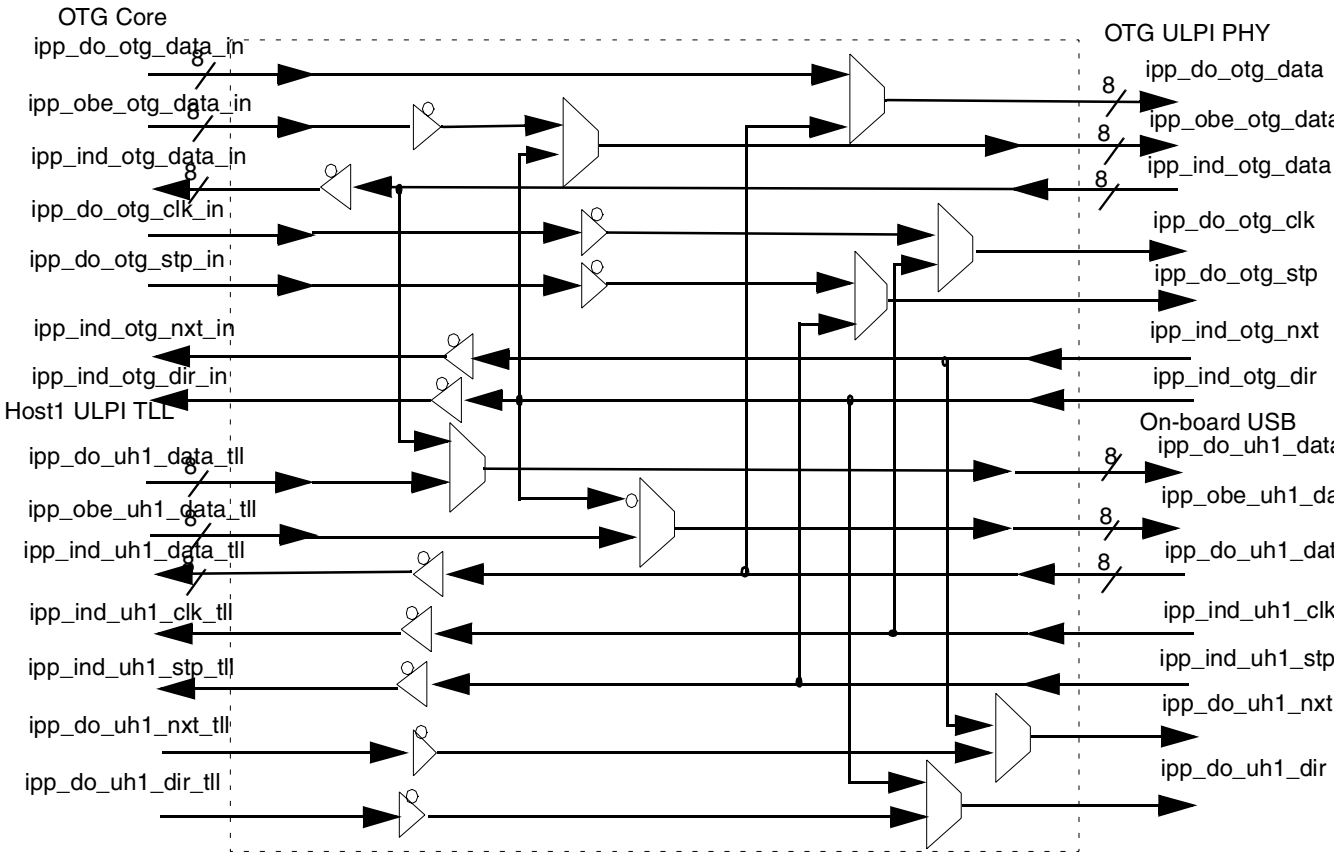


Figure 60-16. ULPI Bypass Pin Route Function Diagram

60.3.8 ULPI/Serial MUX

Host1, Host2, and OTG cores can be configured by software for ULPI or Serial PHY operation. The ULPI/Serial mux selects between ULPI interface signals and Serial PHY interface signals. The mux is controlled by the PHY Select signals from the USB core and is switched when the software selects the interface mode.

The default configuration for the mux is Serial mode. Switching to ULPI mode is done by writing the Parallel Transceiver Select (PTS) bits in the PORTSC register with 0b10.

60.3.9 Interrupts

This section discusses the USB core and wake-up interrupts.

60.3.9.1 USB Core interrupts

Each USB core uses one dedicated vector in the interrupt table. The vector numbers associated with each of the cores can be found in the Interrupt section.

With the exception of the wake-up interrupts, all of the interrupt sources are controlled in the USB Cores. Please refer to the USB Core documentation for details.

60.3.9.2 USB Wake-Up Interrupts

Each USB core has an associated wake-up interrupt. The wake-up interrupts are generated outside the USB cores' but use the same vector as the corresponding cores' interrupt. These interrupt are generated by the Power Control Modules which run on the 32 KHz standby clock. The wake-up interrupt is designed to work even when the USB and CPU clocks are disabled, such that a wake-up condition on the USB bus can re-activate the CPU clocks. Therefore, this interrupt request propagates thru combinational logic to the CPU's interrupt module.

Because the wake-up interrupt is generated and cleared on a 32 KHz clock, this interrupt request will respond very slowly to clear actions. For this reason, the software must disable the wake-up interrupt to clear the request flag. Disabling the interrupt will mask the request instantaneously as this is clocked by the CPU clock. The software should than wait for at least three 32-KHz clock cycles before re-enabling this interrupt to allow sufficient time for the request flag to clear. As this interrupt is only used during low-power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to enter USB suspend mode.

60.4 Initialization/Application Information

This section described the detailed application knowledge for Host1, Host2 and OTG ports. It can be generally divided in two parts, one is for host and the other is for device. Host part is applied to three ports; device part is only applied to OTG port. In the following register description, device-related content is for OTG and host-related content is for all three ports.

60.4.1 Software Model

The Device API provides a framework of routines to control the USB-HS OTG High-Speed USB On-The-Go peripheral in USB device applications. It includes an application to respond to the Chapter 9 device framework commands issued by a USB host.

The USB-HS OTG High-Speed USB On-The-Go Device API is designed to significantly simplify the software tasks required to develop a USB device application. The API presents a high-level data transfer interface to the user's application code. All the register, interrupt and DMA interactions with the USB-HS OTG High-Speed USB On-The-Go core are managed by the API. The API also includes routines that handle all the USB device framework commands that are required for all USB devices.

The Host Stack provides a layered software architecture to control all aspects of a USB bus system. The Host Controller Device (HCD) interface controls the functions of an embedded EHCI host controller. The USB driver layer provides all the USB driver functions to enumerate, manage and schedule a USB bus system, while the upper layers of the stack support standard USB device class interfaces to the device drives running on the embedded system.

For details on the HS-USB OTG High-Speed USB On-The-Go Software Stack refer to following the documentation provided with the software products.

- **ANSI-C OTG software Stack provides Host and Device application support.** USB software included with the HS-USB OTG High-Speed USB On-The-Go core is tested with the hardware.
- **OTG Application Program Interface (API) handles OTG protocols.** Connect and disconnect events are handled as well as the OTG Host Negotiation Protocol (HNP) and Session Request Protocol (SRP) state machines. The OTG code calls the Host or Device API functions based on the connection state of the OTG state machines.
- **Host API to speed up Host software development.** Simple API calls allow direct interaction with USB pipes. Additional layers support bus enumeration, bus management and a growing set of supported USB classes.
- **Device API to speed up peripheral development.** USB peripheral characteristics such as endpoints, configurations, interfaces, and alternate settings are controlled by supplied ANSI-C firmware. Chapter 9 Device Frame work command set reduces software development time. Simple API interface allows quick coding of USB device applications.

60.4.1.1 Device Data Structure

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus, as shown in Figure 60-17. Using a set of linked list **transfer descriptors**, pointed to by a **queue head**, the device controller will perform the data transfers.

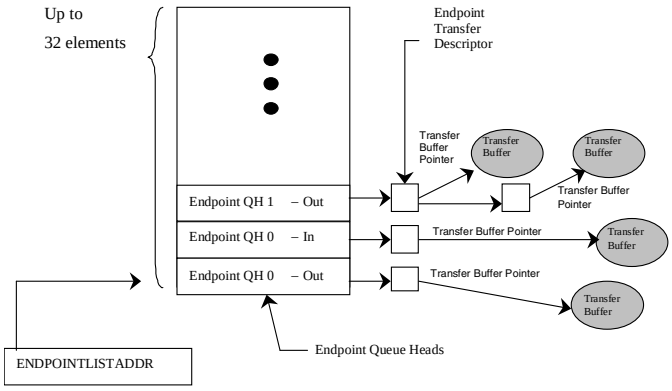


Figure 60-17. End Point Queue Head Organization

The HS-USB OTG High-Speed USB On-The-Go Device API incorporates and abstracts for the application developer all of the information contained in the device operational model.

60.4.1.2 Host Data Structure

The host data structures are used to communicate control, status, and data between software and the Host Controller, as shown in Figure 60-18 and Figure 60-19. The **Periodic Frame List** is an array of pointers for the periodic schedule. A sliding window on the Periodic Frame List is used. The **Isochronous Transfer List** is where all the control and bulk transfers are managed. The HS-USB OTG High-Speed USB On-The-Go Host API incorporates and abstracts for the application developer all of the information contained in the host operational model.

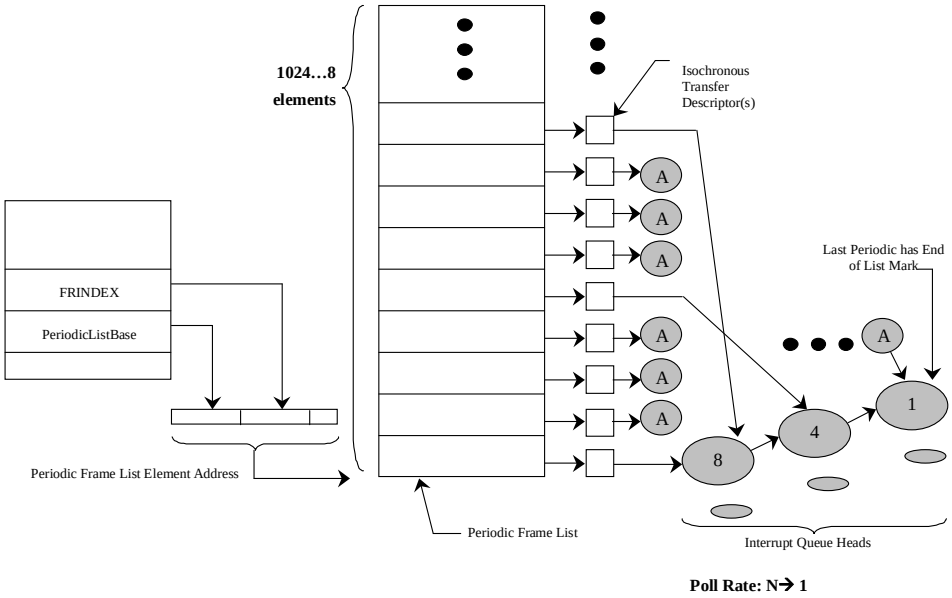


Figure 60-18. Periodic Schedule Organization

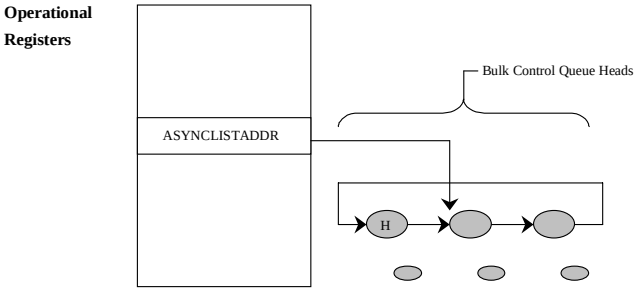


Figure 60-19. Asynchronous Schedule Organization

60.4.2 Register Interface

Slave accesses from the controlling processor enables access to the configuration, control, and status registers. One function of the system address map is the registers base address, which must begin on a **DWord** (32-bit) boundary. Register offset definitions are listed in Table 60-26.

Configuration, control and status registers are divided into three categories, identification, capability and operational registers.

- Identification registers are used to declare the slave interface presence along with the complete set of the hardware configuration parameters.
- Static, read only capability registers define the software limits, restrictions, and capabilities of the host/device controller.
- Operational registers are comprised of dynamic control or status registers that may be read only, read/write, or read/write to clear. The following sections define the use of these registers.

EHCI registers are listed alongside device registers to show the complementary nature of host and device control.

NOTE

Host mode EHCI compatibility begins at offset 0x100. If it is necessary to begin the EHCI register set at offset 0x000, the identification registers are disabled from the address map by connecting the upper most address bit of the slave interface to a logic level '1' and adjusting the offsets below accordingly.

Table 60-26. Interface Register Sets

Offset	Register Set	Explanation
000h to 0FCh	Identification Registers	Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.
100h to 124h	Capability Registers	Capability registers specify the limits, restrictions, and capabilities of a host/device controller implementation. These values are used as parameters to the host/device controller driver.
140h to 1FCh	Operational Registers	Operational registers are used by the system software to control and monitor the operational state of the host/device controller.

60.4.2.1 Configuration, Control and Status Register Set

Table 60-27 shows the device/host capability registers.

Table 60-27. Device/Host Capability Registers

Offset	Size (bytes)	Mnemonic	Register Name	DEV	OTG	SPH	MPH
000h	4	ID	Identification Register	÷	÷	÷	÷
004h	4	HWGENERAL	General Hardware Parameters	÷	÷	÷	÷
008h	4	HWHOST	Host Hardware Parameters		÷	÷	÷
00Ch	4	HWDEVICE	Device Hardware Parameters	÷	÷	—	—

Table 60-27. Device/Host Capability Registers (continued)

Offset	Size (bytes)	Mnemonic	Register Name	DEV	OTG	SPH	MPH
010h	4	HWTXBUF	TX Buffer Hardware Parameters	÷	÷	÷	÷
014h	4	HWRXBUF	RX Buffer Hardware Parameters	÷	÷	÷	÷
018h	4	HWTTTXBUF	TT-TX Buffer Hardware Parameters				÷
01Ch	4	HWTTRXBUF	TT-RX Buffer Hardware Parameters				÷
020h-0FCh	232	Reserved	N/A				
080h	4	GPTIMER0LD	General Purpose Timer #0 Load Register				
084h	4	GPTIMER0CTRL	General Purpose Timer #0 Control Register				
088h	4	GPTIMER1LD	General Purpose Timer #1 Load Register				
08ch	4	GPTIMER1CTRL	General Purpose Timer #1 Control Register				
100h	1	CAPLENGTH	Capability Register Length	÷	÷	÷	÷
101h	1	Reserved	N/A				
102h	2	HCIVERSION	Host Interface Version Number		÷	÷	÷
104h	4	HCSPARAMS	Host Ctrl. Structural Parameters		÷	÷	÷
108h	4	HCCPARAMS	Host Ctrl. Capability Parameters		÷	÷	÷
10Ch – 11Fh	20	Reserved	N/A				
120h	2	DCIVERSION	Dev. Interface Version Number	÷	÷		
122h	2	Reserved	N/A	÷			
124h	4	DCCPARAMS	Device Ctrl. Capability Parameters	÷	÷		
128h – 13Ch	24	Reserved	N/A				

Table 60-27. Device/Host Capability Registers (continued)

Offset	Size (bytes)	Mnemonic	Register Name	DEV	OTG	SPH	MPH
140h	4	USBCMD	USB Command	÷	÷	÷	÷
144h	4	USBSTS	USB Status	÷	÷	÷	÷
148h	4	USBINTR	USB Interrupt Enable	÷	÷	÷	÷
14Ch	4	FRINDEX	USB Frame Index	÷	÷	÷	÷
150h	4	Reserved	4G Segment Selector				
154h	4	PERIODICLISTBASE	Frame List Base Address		÷	÷	÷
		Device Addr	USB Device Address	÷	÷		
158h	4	ASYNCLISTADDR	Next Asynchronous List Address		÷	÷	÷
		Endpointlist Addr	Address at Endpoint list in memory	÷	÷		
15Ch	4	ASYNCTTSTS	Asynchronous Buffer Status For Embedded TT.				÷
160h	4	BURSTSIZE	Programmable Burst Size	÷	÷	÷	÷
164h	4	TXFILLTUNING	Host Transmit Pre-Buffer Packet Tuning		÷	÷	÷
168h	4	TXTTFILLTUNING	Host TT Transmit Pre-Buffer Packet Tuning				÷
16Ch	4	IC_USB	IC_USB enable and voltage negotiation				
170-17Ch	16	N/A	Reserved				
180h	4	CONFIGFLAG	Configured Flag Register		÷	÷	÷
184h	4	PORTSC1	Port Status/Control 1	÷	÷	÷	÷
188h	4	PORTSC2	Port Status/Control 2				÷
...	4	PORTSCx	Port Status/Control x				÷
1A0h	4	PORTSC8	Port Status/Control 8				÷
1A4h	4	OTGSC	On-The-Go (OTG) Status and Control		÷		
1A8h	4	USBMODE	USB Device Mode	÷	÷	÷	÷
1ACh	4	ENPDTSETUPSTAT	Endpoint Setup Status	÷	÷		

Table 60-27. Device/Host Capability Registers (continued)

Offset	Size (bytes)	Mnemonic	Register Name	DEV	OTG	SPH	MPH
1B0h	4	ENDPTPRIME	Endpoint Initialization	÷	÷		
1B4h	4	ENDPTFLUSH	Endpoint De-Initialize	÷	÷		
1B8h	4	ENDPTSTATUS	Endpoint Status	÷	÷		
1BCh	4	ENDPTCOMPLETE	Endpoint Complete	÷	÷		
1C0h	4	ENDPTCTRL0	Endpoint Control 0	÷	÷		
1C4h	4	ENDPTCTRL1	Endpoint Control 1	÷	÷		
…	4	ENDPTCTRLx	Endpoint Control x	÷	÷		
1FCh	4	ENDPTCTRL15	Endpoint Control 15	÷	÷		
Note: *Italic* text indicates a deviation from EHCI for device.							

60.4.2.2 Summary of Register Layouts

Table 60-27 shows the HS-USB register summary.

Table 60-28. HS-USB Register Summary

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
000h ID	reserved								REVISION								NID								ID							
004h HWGENERAL	reserved																								SPHM	SPHW	PHYW	BWT	CLKC	RT		
008h HWHOST	TTPER										TTASY						reserved													NPORT		HC
00Ch HWDEVICE	reserved																										DEVP					DC
010h HWTXBUF	TXLC	reserved							TXCHANADD								TXADD								TXBURST							
014h HWRXBUF	reserved																RXADD								RXBURST							
020h Reserved	reserved																															
… Reserved	reserved																															
0FCh Reserved	reserved																															
100h CAPLENGTH																									CAPLENGTH							

Table 60-28. HS-USB Register Summary (continued)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
101h Reserved																									reserved							
102h HCIVERSION																	HCIVERSION															
104h HCSPARAMS	reserved				N_TT				N_PTT				reserved			PI	N_CC				N_PCC		reserved		PPC			N_PORTS			
108h HCCPARAMS	reserved																EEC{[7:0]						IST[7:4]				R	R	PFL	ADC	
10Ch Reserved	reserved																															
… Reserved	reserved																															
11Fh Reserved	reserved																															
120h DCIVERSION																	DCIVERSION															
122h Reserved																	reserved															
124h DCCPARAMS	reserved																							HDC	DC	R	R	DEN				
128h Reserved	reserved																															
… Reserved	reserved																															
13Ch Reserved	reserved																															
140h USBCMD	reserved								ITC								FS2	R	SUTW	ATDTW	ASPE	R	ASP1	ASP0	IAA	ASE	PSE	FS1	FS0	RST	RS	
144h USBSTS	reserved																AS	PS	RCL	HCH	reserved		SLRI	URI	AAI	SEI	FRI	PCI	UEI	UI		
148h USBINTR	reserved																						SLRE	URE	AAE	SEE	FRE	PCE	UEE	UE		
14Ch FRINDEX	reserved																FRINDEX[13:0]															
150h Reserved	reserved																															

Table 60-28. HS-USB Register Summary (continued)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
154h PERIODICLISTBASE	PERBASE[31:12]																					reserved										
Device Addr	USBADR[31:25]							reserved																								
158h ASYNCLISTADDR	ASYBASE[31:5]																											reserved				
Endpointlist Addr	EPBASE[31:11]																					reserved										
15Ch ASYNCTTSTS	reserved																														TTAC	TTAS
160h BURSTSIZE	reserved																TXPBURST								RXPBURST							
164h TXFILLTUNING	reserved										TXFIFOTHRES						R			TXSCHHEALTH				TXSCHOH							
168h TXTTFILLTUNING	reserved																			TXTTSCHHEALTH		R		TXTTSCHOH							
16Ch IC_USB	IC8	IC_VDD8			IC7	IC_VDD7			IC6	IC_VDD6			IC5	IC_VDD6			IC4	IC_VDD4			IC3	IC_VDD3		IC2	IC_VDD2			IC1	IC_VDD1			
170 ULPI Viewport																																
174 Reserved	reserved																															
… Reserved	reserved																															
17Ch Reserved	reserved																															
180h CONFIGFLAG	set to zero																															1
184h PORTSC1	PTS		STS	PTW	PSPD		R	PFSC	PHCD	WKOC	WKDS	WKCN	PTC			PIC		PO	PP	LS		SUSP/FPR		OCC	OCA	PEC	PE	CSC	CCS		

Table 60-28. HS-USB Register Summary (continued)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
188h PORTSC2	PTS		STS	PTW	PSPD		R	PFSC	PHCD	WKOC	WKDS	WKCN	PTC				PIC		PO	PP	LS		SUSP/HSP		FPR	OCC	OCA	PEC	PE	CSC	CCS
… PORTSCx	PTS		STS	PTW	PSPD		R	PFSC	PHCD	WKOC	WKDS	WKCN	PTC				PIC		PO	PP	LS		SUSP/HSP		FPR	OCC	OCA	PEC	PE	CSC	CCS
1A0h PORTSC8	PTS		STS	PTW	PSPD		R	PFSC	PHCD	WKOC	WKDS	WKCN	PTC				PIC		PO	PP	LS		SUSP/HSP		FPR	OCC	OCA	PEC	PE	CSC	CCS
1A4h OTGSC	R	DPIE	1msE	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	R	DPIS	1msS	BSEIS	BSVIS	ASVIS	AVVIS	IDIS	R	DPS	1msT	BSE	BSV	ASV	reserved			DP	OT	R		VC	VD
1A8h USBMODE	reserved																										SDIS	SLOM	ES		CM	
1ACh ENPDTSETUPSTAT	reserved																ENDPTSETUPSTAT															
1B0h ENDPTPRIME	PETB[15:0]																PERB[15:0]															
1B4h ENDPTFLUSH	FETB[15:0]																FERB[15:0]															
1B8h ENDPTSTATUS	ETBR[15:0]																ERBR[15:0]															
1BCh ENDPTCOMPLETE	ETCE[15:0]																ERCE[15:0]															
1C0h ENDPTCTRL0	reserved								TXE	reserved			TXT		R	TXS	reserved						RXE	reserved				RXT		R	RXS
1C4h ENDPTCTRL1	reserved								TXE	TXR	TXI	R	TXT		TXD	TXS	reserved						RXE	RXR		RXI	R	RXT		RXD	RXS
… ENDPTCTRLx	reserved								TXE	TXR	TXI	R	TXT		TXD	TXS	reserved						RXE	RXR		RXI	R	RXT		RXD	RXS
1FCh ENDPTCTRL15	reserved								TXE	TXR	TXI	R	TXT		TXD	TXS	reserved						RXE	RXR		RXI	R	RXT		RXD	RXS

60.4.2.3 Identification Registers

Identification registers are used to declare the slave interface presence and to include a table of the hardware configuration parameters.

60.4.2.3.1 ID

The Identification register (ID) provides a simple way to determine if the USB-HS OTG High-Speed USB On-The-Go USB 2.0 core is provided in the system. The ID register identifies the USB-HS OTG High-Speed USB On-The-Go USB 2.0 core and its revision.

Figure 60-20 shows the identification register.

Address: Base + 000h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

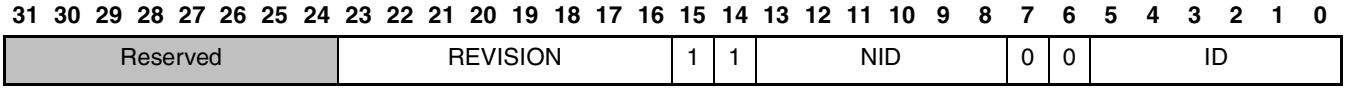


Figure 60-20. ID—Identification Register

The register fields are described in Table 60-29.

Table 60-29. Identification Register Field Descriptions

Field	Description
ID[5:0]	**Configuration number.** This number is set to 0x05 and indicates that the peripheral is the USB-HS OTG High-Speed USB On-The-Go USB 2.0 core.
NID[5:0]	Ones complement version of ID[5:0].
REVISION[7:0]	Revision number of the core
Reserved	These bits are reserved and should be set to zero

60.4.2.3.2 HWGENERAL

The HWGeneral register, shown in Figure 60-21, is for the general hardware parameters as defined in system-level issues and core configuration.

Address: Base + 004h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

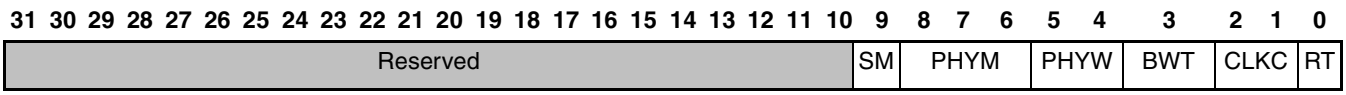


Figure 60-21. HWGENERAL—General Hardware Parameters

The register fields are described in Table 60-30.

Table 60-30. HWGeneral Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
SM	VUSB_HS_PHY_SERIAL
PHYM	VUSB_HS_PHY_TYPE
PHYW	VUSB_HS_PHY16_8
BWT	Reserved for internal testing.
CLKC	VUSB_HS_CLOCK_CONFIGURATION
RT	VUSB_HS_RESET_TYPE

60.4.2.3.3 HWHOST

Host hardware parameters as defined in system-level issues and core configuration.

Address: Base + 008h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

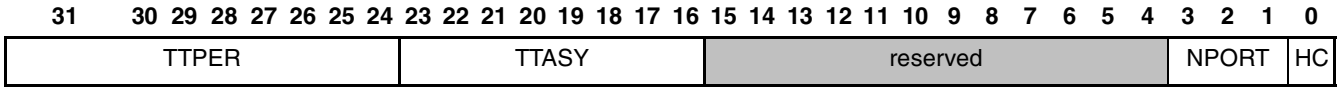


Figure 60-22. HWHOST–Host Hardware Parameters

The register fields are described in Table 60-31.

Table 60-31. HWHost Field Descriptions

Field	Description
TTPER	VUSB_HS_TT_PERIODIC_CONTEXTS
TTASY	VUSB_HS_TT_ASYNC_CONTEXTS
reserved	**Reserved.** These bits are reserved and should be set to zero.
NPORT	VUSB_HS_NUM_PORT-1
HC	VUSB_HS_HOST

60.4.2.3.4 HWDEVICE

Device hardware parameters as defined in System Level Issues and Core Configuration.

Address: Base + 00Ch

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

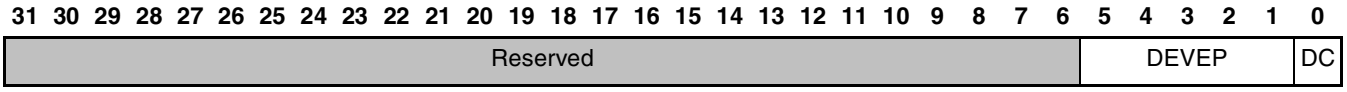


Figure 60-23. HWDEVICE – Device Hardware Parameters

The register fields are described in Table 60-32.

Table 60-32. HWDevice Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
DEVEP	VUSB_HS_DEV_EP
DC	device capable; [VUSB_HS_DEV /= 0]

60.4.2.3.5 HWTXBUF

TX buffer hardware parameters as defined in system-level issues and core configuration.

Address: Base + 010h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

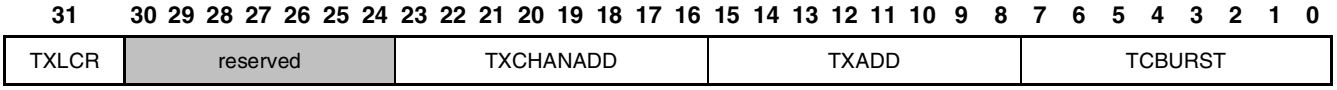


Figure 60-24. HWTXBUF – TX Buffer Hardware Parameters

The register fields are described in Table 60-33.

Table 60-33. HWTXBUF Field Descriptions

Field	Description
TXLC	VUSB_HS_TX_LOCAL_CONTEXT_REGISTERS
reserved	**Reserved.** These bits are reserved and should be set to zero.
TXCHANADD	VUSB_HS_TX_CHAN_ADD
TXADD	VUSB_HS_TX_ADD
TCBURST	VUSB_HS_TX_BURST

60.4.2.3.6 HWRXBUF

RX buffer hardware parameters as defined in System Level Issues and Core Configuration.

Address: Base + 014h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

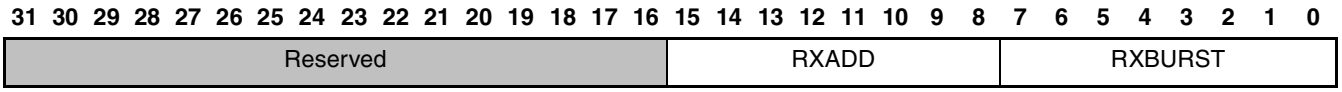


Figure 60-25. WRXBUF – RX Buffer Hardware Parameters

The register fields are described in Table 60-34.

Table 60-34. WRXBUF Field Descriptions

Field	Description
reserved	**Reserved.** These bits are reserved and should be set to zero.
RXADD	VUSB_HS_RX_ADD
RXBURST	VUSB_HS_RX_BURST

60.4.2.4 Device/Host Capability Registers

Device/Host capability registers specify the software limits, restrictions, and capabilities of the host/device controller implementation.

60.4.2.4.1 CAPLENGTH – EHCI Compliant

This register is used to indicate which offset to add to the register base address at the beginning of the Operational Register.

Address: Base + 100h

Default Value: 40h

Attribute: Read Only

Size: 8 bits

Figure 60-26. CAPLENGTH – Capability Register Length

60.4.2.4.2 HCIVERSION – EHCI Compliant

This is a two-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Address: Base + 102h

Default Value: 0100h

Attribute: Read Only

Size: 16 bits

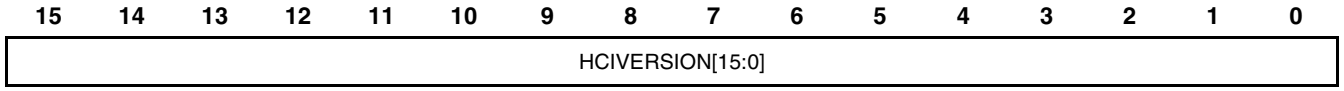


Figure 60-27. HCIVERSION – Host Interface version number

60.4.2.4.3 HCSPARAMS – EHCI Compliant with extensions

Port steering logic capabilities are described in this register.

Address: Base + 104h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 32 bits

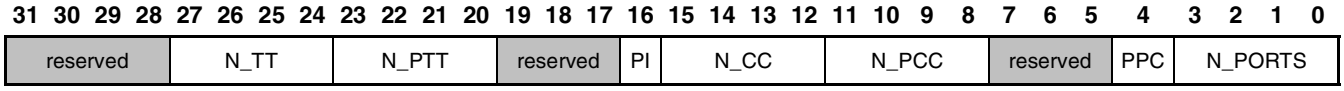


Figure 60-28. HCSPARAMS – Host Control Structural Parameters

The register fields are described in Table 60-35.

Table 60-35. HCSPARAMS Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
PI	**Port Indicators (P INDICATOR).** This bit indicates whether the ports support port indicator control. When set to one, the port status and control registers include a read/writeable field for controlling the state of the port indicator. This field will always be "1".
N_TT[3:0]	**Number of Transaction Translators (N_TT).** This field indicates the number of embedded transaction translators associated with the USB2.0 host controller. For Multi-Port Host this field will always equal “0001”. For all other implementations, N_TT = “0000”. This in a non-EHCI field to support embedded TT.
N_PTT[3:0]	**Number of Ports per Transaction Translator (N_PTT).** This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. For Multi-Port Host this field will always equal N_PORTS. For all other implementations, N_PTT = “0000”. This in a non-EHCI field to support embedded TT.
N_CC[3:0]	**Number of Companion Controller (N_CC).** This field indicates the number of companion controllers associated with this USB2.0 host controller. A zero in this field indicates there are no internal Companion Controllers. Port-ownership hand-off is not supported. A value larger than zero in this field indicates there are companion USB1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports. In this implementation this field will always be "0".

Table 60-35. HCSPARAMS Field Descriptions

N_PCC[3:0]	**Number of Ports per Companion Controller.** This field indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC. In this implementation this field will always be "0".
PPC	**Port Power Control.** This field indicates whether the host controller implementation includes port power control. A one indicates the ports have port power switches. A zero indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register. This field will always be "0" for a device only implementation.
N_PORTS[3:0]	**Number of downstream ports.** This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register. Valid values are in the range of 1h to Fh. A zero in this field is undefined. The number of ports for a host implementation is parameterizable from 1 to 8. This field will always be 1 for device only implementation.

60.4.2.4.4 HCCPARAMS – EHCI Compliant

This register identifies multiple mode control (time-base bit functionality) addressing capability.

Address: Base + 108h

Default Value: 0006h

Attribute: Read Only

Size: 32 bits

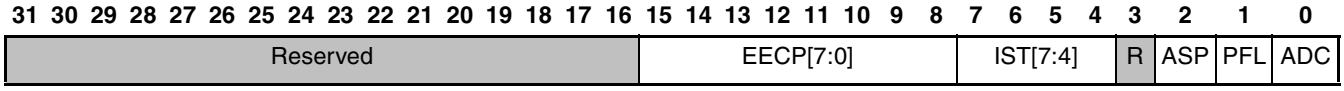


Figure 60-29. HCCPARAMS – Host Control Capability Parameters

The register fields are described in Table 60-36.

Table 60-36. HCCPARAMS Field Descriptions

Field	Description
reserved	**reserved.** These bits are reserved and should be set to zero.
EECP[7:0]	**EHCI Extended Capabilities Pointer.** Default = 0. This optional field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device. For this implementation this field is always "0".

Table 60-36. HCCPARAMS Field Descriptions

IST[7:4]	**Isochronous Scheduling Threshold.** Default = implementation dependent. This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame. This field will always be "0".
R	**Reserved.** These bits are reserved and should be set to zero.
ASP	**Asynchronous Schedule Park Capability.** Default = 1. If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USBCMD register. This field will always be "1"
PFL	**Programmable Frame List Flag.** If this bit is set to zero, then the system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and must be set to zero. If set to a one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous. This field will always be "1".
ADC	64-bit Addressing Capability. This field will always be "0". No 64-bit addressing capability is supported.

60.4.2.4.5 DCIVERSION (Non-EHCI)

The device controller interface conforms to the two-byte BCD encoding of the interface version number contained in this register.

Address: Base + 120h

Default Value: Implementation Dependent

Attribute: Read Only

Size: 16 bits

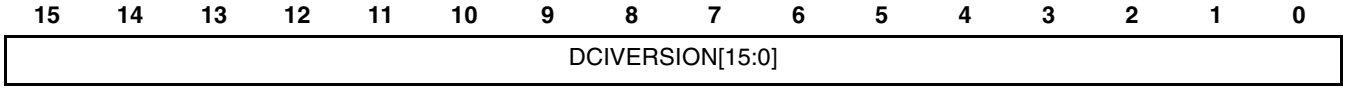


Figure 60-30. DCIVERSION – Device Interface Version Number

60.4.2.4.6 DCCPARAMS (Non-EHCI)

These fields describe the overall host/device capability of the controller.

Address: Base + 124h

Default Value: Implementation Dependant

Attribute: Read Only

Size: 32 bits

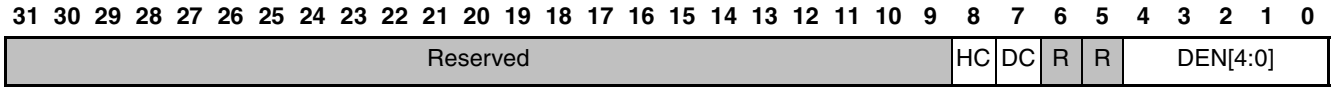


Figure 60-31. DCCPARAMS - Device Control Capability Parameters

The register fields are described in Table 60-37.

Table 60-37. DCCPARAMS Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
HC	**Host Capable.** When this bit is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
DC	**Device Capable.** When this bit is 1, this controller is capable of operating as a USB 2.0 device.
R	**Reserved.** These bits are reserved and should be set to zero.
DEN[4:0]	**Device Endpoint Number.** This field indicates the number of endpoints built into the device controller. If this controller is not device capable, then this field will be zero. Valid values are 0–16.

60.4.2.5 Device/Host Timer Registers (Non-EHCI)

The host/device controller drivers can measure time related activities using these timer registers. These registers are not part of the standard EHCI controller.

60.4.2.5.1 GPTIMER0LD (Non-EHCI)

This register contains the timer duration or load value. See the GPTIMER0CTRL (Non-EHCI) for a description of the timer functions.

Address: Base + 80h

Default Value: 00000000h

Attribute: Read/Write

Size: 32 bits

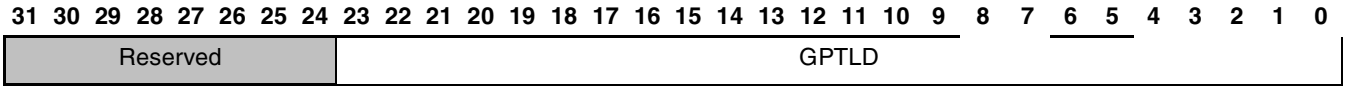


Figure 60-32. GPTIMER0LD-General Purpose Timer #0 Load Register

The register fields are described in Table 60-38.

Table 60-38. GPTIMER0LD Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
H\GPTLD	**General Purpose Timer Load Value.** This field is the value to be loaded into the GPTCNT countdown timer on a reset action. This value represents the time in microseconds minus 1 for the timer duration.

60.4.2.5.2 GPTIMER0CTRL (Non-EHCI)

This register contains the control for the timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two modes supported by this timer, the first is a one shot and the second is a looped count which is described in the register table below. When the timer counter value transitions to zero, an interrupt can be generated though the use of the timer interrupts in the USBSTS and USBINTR registers.

Address: Base + 84h

Default Value: 00000000h

Attribute: Read Only, Write Only, Read/Write

Size: 32 bits

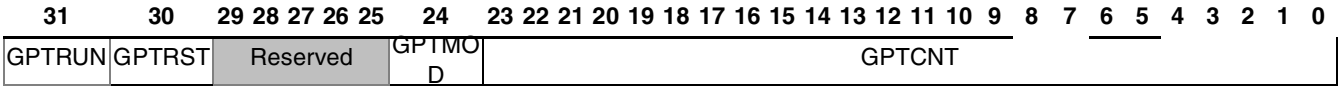


Figure 60-33. GPTIMER0LD—General Purpose Timer #0 Controller

The register fields are described in Table 60-39.

Table 60-39. GPTIMER0LD

Field	Description
GPTRUN	**General Purpose Timer Run (0)-Timer Stop; (1)-Timer Run.** This bit enable the GPT to run. Setting or Clearing this bit will not have an effect on the GPTCNT except.
GPTRST	**General Purpose Timer Reset (0)-No action; (1)-Load Counter Value**. This bit will reload the GPTCNT with the value in GPTLD.
Reserved	**Reserved.** These bits are reserved and should be set to zero.
GPTMOD	**General Purpose Timer Mode.** (0)-One Shot; (1)-Repeat.In one-shot mode the timer will count to zero, generate an interrupt and stop until the timer is reset by software. In repeat mode the timer will count to zero, generate an interrupt and automatically reload the counter to begin again.
GPTCNT	**General Purpose Timer Counter**. This field is the value of running timer.

60.4.2.5.3 GPTIMER1LD (Non-EHCI)

Same as GPTIMER0LD.

Address: Base + 88h

Default Value: 00000000h

High-Speed USB Controller Core Reference

Attribute: Read/Write

Size: 32 bits

60.4.2.5.4 GPTIMER1CTRL (Non-EHCI)

Same as GPTIMER0CTRL.

Address: Base + 8Ch

Default Value: 00000000h

Attribute: Read Only, Write Only, Read/Write

Size: 32 bits

60.4.2.6 Device/Host Operational Registers

Operational registers are comprised of dynamic control or status registers that may be read only, read/write, or read/write to clear. The following sections define the use of these registers.

60.4.2.6.1 USBCMD

The serial bus host/device controller executes the command indicated in this register.

Address: Base + 140h

Default Value: 00080B00h (host mode)

00080000h (device mode)

Attribute: Read Only, Read/Write, Write Only (field dependent)

Size: 32 bits

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved																ITC[7:0]								FS2	R	SUTW	ATDTW	ASPE	R	ASP1	ASP0	LR	IAA	ASE	PSE	FS1	FS0	RST	RS

Figure 60-34. USBCMD – USB Command Register

The register fields are described in Table 60-40.

Table 60-40. USBCMD Field Descriptions

Field	Description
R, reserved	**Reserved.** These bits are reserved and should be set to zero.
ITC[7:0]	**Interrupt Threshold Control —Read/Write.** Default 08h. The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below. Value Maximum Interrupt Interval 00h Immediate (no threshold) 01h 1 micro-frame 02h 2 micro-frames 04h 4 micro-frames 08h 8 micro-frames 10h 16 micro-frames 20h 32 micro-frames 40h 64 micro-frames

Table 60-40. USBCMD Field Descriptions (continued)

SUTW	**Setup TripWire – Read/Write.** [device mode only] This bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (See USBCMD) then there exists a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software and will be cleared by hardware when a hazard exists. For more information on the use of this bit, see the Device Operational Model section of the USB-HS OTG High-Speed USB On-The-Go DEV reference manual.
ATDTW	**Add dTD TripWire – Read/Write.** [device mode only] This bit is used as a semaphore to ensure the to proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software. This bit shall also be cleared by hardware when is state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized. For more information on the use of this bit, see the Device Operational Model section of the USB-HS OTG High-Speed USB On-The-Go DEV reference manual.
ASPE	**Asynchronous Schedule Park Mode Enable (OPTIONAL)** — **Read/Write.** If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled. This field is set to "1" in this implementation.
ASP[1:0]	**Asynchronous Schedule Park Mode Count (OPTIONAL)** — **Read/Write**. If the *Asynchronous Park Capability* bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is RO. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. See Section 4.10.3.2 for full operational details. Valid values are 1h to 3h. Software must not write a zero to this bit when *Park Mode Enable* is a one as this will result in undefined behavior. This field is set to 3h in this implementation.
LR	**Light Host/Device Controller Reset (OPTIONAL) — Read Only.** Not Implemented. This field will always be "0".
IAA	**Interrupt on Async Advance Doorbell — Read/Write.** This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold. The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results. This bit is only used in host mode. Writing a one to this bit when device mode is selected will have undefined results.
ASE	**Asynchronous Schedule Enable — Read/Write.** Default 0b. This bit controls whether the host controller skips processing the Asynchronous Schedule. Valuesmeaning 0 Do not process the Asynchronous Schedule. 1 Use the ASYNCLISTADDR register to access theAsynchronous Schedule. Only the host controller uses this bit.
PSE	**Periodic Schedule Enable— Read/Write.** Default 0b. This bit controls whether the host controller skips processing the Periodic Schedule. Values meaning 0 Do not process the Periodic Schedule 1 Use the PERIODICLISTBASE register to access the PeriodicSchedule. Only the host controller uses this bit.

Table 60-40. USBCMD Field Descriptions (continued)

FS[2:0]	**Frame List Size — (Read/Write or Read Only).** Default 000b. This field is Read/Write only if Programmable Frame List Flag in the HCCPARAMS registers is set to one. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. Note that this field is made up from USBCMD bits 15, 3 and 2. Values meaning 000 1024 elements (4096 bytes) Default value 001 512 elements (2048 bytes) 010 256 elements (1024 bytes) 011 128 elements (512 bytes) 100 64 elements (256 bytes) 101 32 elements (128 bytes) 110 16 elements (64 bytes) 111 8 elements (32 bytes) Only the host controller uses this field.
RST	**Controller Reset (RESET) — Read/Write.** Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register. Host Controller: When software writes a one to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior. Device Controller: When software writes a one to this bit, the Device Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Writing a one to this bit when the device is in the attached state is not recommended, since the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.
RS	**Run/Stop (RS) – Read/Write.** Default 0b. 1=Run. 0=Stop. Host Controller: When set to a 1, the Host Controller proceeds with the execution of the schedule. The Host Controller continues execution as long as this bit is set to a one. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the host controller is in the Halted state (i.e. HCHalted in the USBSTS register is a one). Device Controller: Writing a one to this bit will cause the device controller to enable a pull-up on D+ and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the device controller has been properly initialized. Writing a 0 to this will cause a detach event.

60.4.2.6.2 USBSTS

This register indicates various states of the Host/Device Controller and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus. Software clears certain bits in this register by writing a 1 to them.

Address: Base + 144h

Default Value: 00001000h (host mode)

00000000h (device mode)

Attribute: Read Only, Read/Write, Read/Write-Clear (field dependant)

Size: 32 bits

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TI1	TI0	Reserved								AS	PS	RCL	HCH	R	ULPII	R	SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI

Figure 60-35. USBSTS – USB Status

The register fields are described in Table 60-41.

Table 60-41. USBSTS Field Descriptions

Field	Description
Reserved	Reserved. These bits are reserved and should be set to zero.
TI1	General Purpose Timer Interrupt 1(GPTINT1)--R/WC. This bit is set when the counter in the GPTIMER1CTRL register transitions to zero, writing a one to this bit will clear it.
TI0	General Purpose Timer Interrupt 0(GPTINT0)--R/WC. This bit is set when the counter in the GPTIMER0CTRL register transitions to zero, writing a one to this bit will clear it.
Reserved	**Reserved.** These bits are reserved and should be set to zero.
AS	**Asynchronous Schedule Status — Read Only.** 0=Default. This bit reports the current real status of the Asynchronous Schedule. When set to zero the asynchronous schedule status is disabled and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0). Only used by the host controller.
PS	**Periodic Schedule Status — Read Only.** 0=Default. This bit reports the current real status of the Periodic Schedule. When set to zero the periodic schedule is disabled, and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0). Only used by the host controller.
RCL	**Reclamation — Read Only.** 0=Default. This is a read-only status bit used to detect an empty asynchronous schedule. Only used by the host controller.
HCH	**HCHalted — Read Only.** 1=Default. This bit is a zero whenever the Run/Stop bit is a one. The Host Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (for example, internal error). Only used by the host controller.

Table 60-41. USBSTS Field Descriptions (continued)

R	**Reserved.** These bits are reserved and should be set to zero.
ULPII	**ULPI Interrupt – R/WC.** 0=Default. When the ULPI Viewport is present in the design, an event completion will set this interrupt. Used by both host & device controller. Only present in designs where configuration constant VUSB_HS_PHY_ULPI = 1.
R	**Reserved.** These bits are reserved and should be set to zero.
SLI	**DCSuspend – R/WC.** 0=Default. When a device controller enters a suspend state from an active state, this bit will be set to a one. The device controller clears the bit upon exiting from a suspend state. Only used by the device controller.
SRI	**SOF Received – R/WC.** 0=Default. When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1ms in device FS mode and every 125ms in HS mode and will be synchronized to the actual SOF that is received. Since the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp. In host mode, this bit will be set every 125us and can be used by host controller driver as a time base. Software writes a 1 to this bit to clear it. This is a non-EHCI status bit.
URI	**USB Reset Received – R/WC.** 0=Default. When the device controller detects a USB Reset and enters the default state, this bit will be set to a one. Software can write a 1 to this bit to clear the USB Reset Received status bit. Only used by the device controller.
AAI	**Interrupt on Async Advance — R/WC.** 0=Default. System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the USBCMD register. This status bit indicates the assertion of that interrupt source. Only used by the host controller.
SEI	**System Error— R/WC.** This bit is will be set to "1" when an Error response is seen to a read on the system interface.
FRI	**Frame List Rollover — R/WC.** The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example. If the frame list size (as programmed in the Frame List Size field of the USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [1 3] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FHINDEX [12] toggles. Only used by the host controller.
PCI	**Port Change Detect — R/WC.** The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port. The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit and the DCSuspend bits respectively. This bit is not EHCI compatible.

Table 60-41. USBSTS Field Descriptions (continued)

UEI	**USB Error Interrupt (USBERRINT) — R/WC.** When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set See Section (Reference Host Operation Model: Transfer/Transaction Based Interrupt – i.e. 4.15.1 in EHCI Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com) for a complete list of host error interrupt conditions. See section Device Error Matrix in the USB-HS OTG High-Speed USB On-The-Go DEV reference manual. The device controller detects resume signaling only.
UI	**USB Interrupt (USBINT) — R/WC.** This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set. This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.

60.4.2.6.3 USBINTR

The interrupts to software are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt is active. The USB Status register (USBSTS) still shows interrupt sources even if they are disabled by the USBINTR register, allowing polling of interrupt events by the software.

Address: Base + 148h

Default Value: 00000000h

Attribute: Read/Write

Size: 32 bits

31 30 29 28 27 26	25	24	23 22 21 20 19 18 17 16 15 14 13 12 11	10	9	8	7	6	5	4	3	2	1	0
Reserved	TIE1	TIE0	Reserved	ULPIE	R	SLE	SRE	URE	AAE	SEE	FRE	PCE	UEE	UE

Figure 60-36. USBINTR – USB Interrupt Enable

The register fields are described in Table 60-42.

Table 60-42. USBINTR Field Descriptions

Field	Interrupt Source	Description
reserved	Reserved.	These bits are reserved and should be set to zero.
TIE1	GPT Interrupt Enable1	When this bit is one and the GPTINT1 in USBSTS register is a one the controller will issue an interrupt. The interrupt is acknowledge by software clear the GPTINT1 bit.
TIE0	GPT Interrupt Enable0	When this bit is one and the GPTINT0 in USBSTS register is a one the controller will issue an interrupt. The interrupt is acknowledge by software clear the GPTINT0 bit.
reserved	Reserved.	These bits are reserved and should be set to zero.
ULPIE	ULPI Enable	When this bit is a one, and the ULPI Interrupt bit in the USBSTS register transitions, the controller will issue and interrupt. The interrupt is acknowledged by software writing a one to the ULPI Interrupt bit. Used by both host & device controller. Only present in designs where configuration constant VUSB_HS_PHY_ULPI = 1.
reserved	Reserved.	These bits are reserved and should be set to zero.

Table 60-42. USBINTR Field Descriptions (continued)

SLE	Sleep Enable	When this bit is a one, and the *DCSuspend* bit in the USBSTS register transitions, the device controller will issue an interrupt. The interrupt is acknowledged by software writing a one to the *DCSuspend* bit. Only used by the device controller.
SRE	SOF Received Enable	When this bit is a one, and the *SOF Received* bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the *SOF Received* bit.
URE	USB Reset Enable	When this bit is a one, and the *USB Reset Received* bit in the USBSTS register is a one, the device controller will issue an interrupt. The interrupt is acknowledged by software clearing the *USB Reset Received* bit. Only used by the device controller.
AAE	Interrupt on Async Advance Enable	When this bit is a one, and the Interrupt on *Async Advance* bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on *Async Advance* bit. Only used by the host controller.
SEE	System Error Enable	When this bit is a one, and the *System Error* bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the *System Error* bit.
FRE	**Frame List Rollover Enable**	When this bit is a one, and the *Frame List Rollover* bit in the USBSTS register is a one, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the *Frame List Rollover* bit. Only used by the host controller.
PCE	Port Change Detect Enable	When this bit is a one, and the *Port Change Detect* bit in the USBSTS register is a one, the host/device controller will issue an interrupt. The interrupt is acknowledged by software clearing the *Port Change Detect* bit.
UEE	USB Error Interrupt Enable	When this bit is a one, and the *USBERRINT* bit in the USBSTS register is a one, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the *USBERRINT* bit in the USBSTS register.
UE	USB Interrupt Enable	When this bit is a one, and the *USBINT* bit in the USBSTS register is a one, the host/device controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the *USBINT* bit.

60.4.2.6.4 FRINDEX

This register is used by the host controller to index the periodic frame list. The register updates every 125 microseconds (once each micro-frame). Bits [N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the USBCMD register.

This register must be written as a DWord. Byte writes produce-undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit. A write to this register while the Run/Stop hit is set to a one produces undefined results. Writes to this register also affect the SOF value.

In device mode this register is read only and, the device controller updates the FRINDEX [13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX [13:3] will be checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] will be set to zero (i.e. SOF

for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] will be increment (i.e. SOF for 125 us micro-frame.)

Address: Base + 14Ch

Default Value: Undefined (free running counter)

Attribute: Read/Write in host mode, Read in device mode

Size: 32 bits

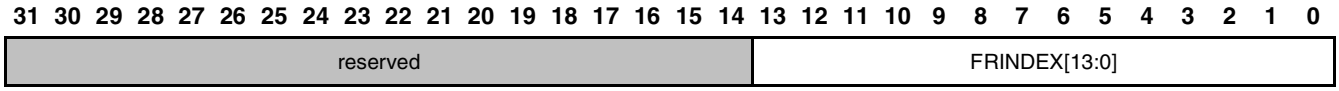


Figure 60-37. FRINDEX – USB Frame Index

The register fields are described in Table 60-43.

Table 60-43. FRINDEX Register Description

Field	Description
reserved	**Reserved.** These bits are reserved and should be set to zero.
FRINDEX	**Frame Index**. The value, in this register, increments at the end of each time frame (for example, micro-frame). Bits [N: 3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register, when used in host mode. USBCMD[Frame List Size] Number Elements N 000b (1024)12 001b (512)11 010b (256)10 011b (128)9 100b (64)8 101b (32)7 110b (16)6 111b (8)5 In device mode the value is the current frame number of the last frame transmitted. It is not used as an index. In either mode bits 2:0 indicate the current microframe.

60.4.2.6.5 CTRLDSSEGMENT

This register is not used in this implementation.

Address: Base + 150h

Default Value: 00000000h

Attribute: Read Only

Size: 32 bits

60.4.2.6.6 PERIODICLISTBASE; DEVICEADDR

This register is shared between the host controller and the device controller operation.

Address: Base + 154h

Default Value: 00000000h

Attribute: Read/Write (Writes must be DWord Writes)

Size: 32 bits

Host Controller (PERIODICLISTBASE)

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

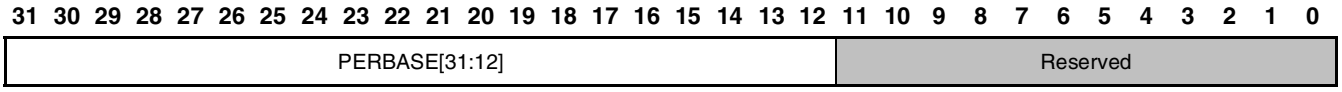


Figure 60-38. PERIODICLISTBASE—Host Controller Frame List Base Address

The register fields are described in Table 60-44.

Table 60-44. PERIODICLISTBASE Register Description

Field	Description
BASEADR	**Base Address (Low).** These bits correspond to memory address signals [31:12], respectively. Only used by the host controller.
reserved	**Reserved.** Must be written as zeros. During runtime, the values of these bits are undefined.

Device Controller (USB DEVICEADDR)

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET_ADDRESS descriptor.

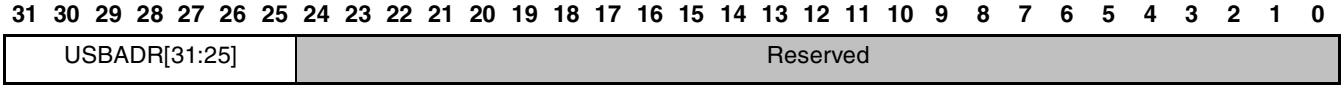


Figure 60-39. DEVICEADDR - Device Controller USB Device Address

The register fields are described in Table 60-45.

Table 60-45. DEVICEADDR Field Descriptions

Field	Description
USBADR	**Device Address.** These bits correspond to the USB device address
reserved	**Reserved.** Must be written as zeros. During runtime, the values of these bits are undefined.

60.4.2.6.7 ASYNCLISTADDR;ENDPOINTLISTADDR

This register is shared between the host controller and the device controller operation.

Address: Base + 158h

Default Value: 00000000h

Attribute: Read/Write (Writes must be DWord Writes)

Size: 32 bits

Host Controller (ASYNCLISTADDR)

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

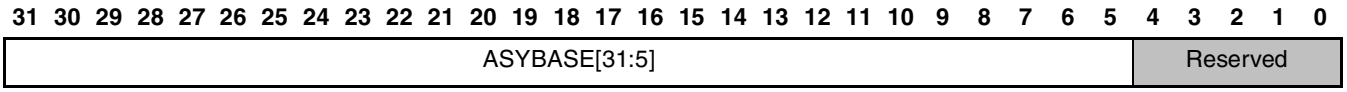


Figure 60-40. ASYNCLISTADDR - Host Controller Next Asynch. Address

The register fields are described in Table 60-46.

Table 60-46. ASYNCLISTADDR Field Descriptions

Field	Description
ASYBASE[31:5]	**Link Pointer Low (LPL).** These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (OH). Only used by the host controller.
Reserved	**Reserved.** These bits are reserved and their value has no effect on operation.

Device Controller (ENDPOINTLISTADDR)

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits [10:0] of this register cannot be modified by the system software and will always return a zero when read.

The memory structure referenced by this physical memory pointer is assumed 64-byte.

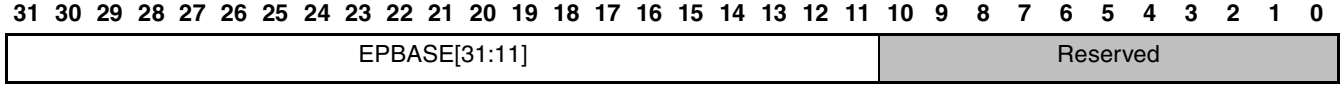


Figure 60-41. ENDPOINTLISTADDR - Device Controller Endpoint List Address

The register fields are described in Table 60-47.

Table 60-47. ENDPOINT Field Descriptions

Field	Description
EPBASE[31:11]	**Endpoint List Pointer(Low).** These bits correspond to memory address signals [31:11], respectively. This field will reference a list of up to 32 Queue Head (OH). (i.e. one queue head per endpoint & direction.)
reserved	**Reserved.** These bits are reserved and their value has no effect on operation.

60.4.2.6.8 BURSTSIZE

This register is used to control dynamically change the burst size used during data movement on the initiator (master) interface.

Address: Base + 160h

Default Value: Implementation Dependent

Attribute: Read/Write (Writes must be DWord Writes)

Size: 32 bits

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved															TXPBURST									RXPBURST							

Figure 60-42. BURSTSIZE—Host Controller Embedded TT Async. Buffer Status

The register fields are described in Table 60-48.

Table 60-48. BURSTSIZE Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and their value has no effect on operation.
TXPBURST	**Programmable TX Burst Length. (Read/Write)** Default is the constant VUSB_HS_TX_BURST. This register represents the maximum length of a the burst in 32-bit words while moving data from system memory to the USB bus.
RXPBURST	**Programmable RX Burst Length. (Read/Write)** Default is the constant VUSB_HS_RX_BURST. This register represents the maximum length of a the burst in 32-bit words while moving data from the USB bus to system memory.

60.4.2.6.9 TXFILLTUNING

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level.

T_s = Total Packet Flight Time (send-only) packet

$T_s = T_0 + T_1$

T_p = Total Packet Time (fetch and send) packet

$T_p = T_{ff} + T_0 + T_1$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure T_p remains before the end of the [micro]frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is < T_s then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a mark will be made the scheduler health counter to note the occurrence of a “back-off” event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Back-offs can be minimized with use of the TSCHHEALTH (T_{ff}) described below.

Address: Base + 164h

Default Value: 00020000h

Attribute: Read/Write (Writes must be DWord Writes)

Size: 32 bits

31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	15 14 13	12 11 10 9 8	7 6 5 4 3 2 1 0
Reserved	TXFIFOTHRES	Reserved	TXSCHEALTH	TXSCHOH

Figure 60-43. TXFILLTUNING

The register fields are described in Table 60-49.

Table 60-49. TXFILLTUNING Field Descriptions

Field	TXFILLTUNING Description
Reserved	**Reserved.** These bits are reserved and their value has no effect on operation.
TXFIFOTHRES	**FIFO Burst Threshold. (Read/Write) [Default = 2]** This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be a low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USBMODE register is set.
TXSCHHEALTH	**Scheduler Health Counter. (Read/Write To Clear) [Default = 0]** This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter and this counter will max. at 31.

Table 60-49. TXFILLTUNING Field Descriptions (continued)

reserved	**Reserved.** These bits are reserved and their value has no effect on operation.
TXSCHOH	**Scheduler Overhead. (Read/Write) [Default = 0]** This register adds an additional fixed offset to the schedule time estimator described above as Tff. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267 µs when a device is connected in High-Speed Mode for OTG & SPH. The time unit represented in this register is 6.333 µs when a device is connected in Low/Full Speed Mode for OTG and SPH. The time unit represented in this register is always 1.267 the MPH product.

60.4.2.6.10 IC_USB

This register enables and controls the IC_USB FS/LS transceiver.

Address: Base + 16Ch

Default Value: 0000h

Attribute: Read/Write

Size: 32 bits

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IC8	IC_VDD8			IC7	IC_VDD7			IC6	IC_VDD6			IC5	IC_VDD5			IC4	IC_VDD4			IC3	IC_VDD_3			IC2	IC_VDD2			IC1	IC_VDD1		

Figure 60-44. IC_USB—IC_USB Enable and Voltage Negotiation

The register fields are described in Table 60-50.

Table 60-50. IC_USB Field Descriptions

Field	Description
IC8, IC7,... IC1	**Inter-Chip transceiver enable.** These bits enables the Inter-Chip transceiver for each port (for MPH case). To enable the interface, the bits PTS must be set to 11 in the PORTSCx. Writing a ‘1’ to each bit selects the IC_USB interface for that port. If the Controller is not MultiPort, IC8 to IC2 will be ‘0’ and Read-Only. If ICx is enabled, IOMUX must set the DP/DM pad into loopback mode.
IC_VDDx	**PIt selects which voltage is being supplied to the peripheral through each port (MPH case)** 000 → No voltage 001 → 1.0 V 010 → 1.2 V 011 → 1.5 V 100 → 1.8 V 101 → 3.0 V 110 → Reserved 111 → Reserved This field is read only and set to ’000’ in case of a device controller. IC_VDD8 to IC_VDD2 are read only and set to ’000’ in case the controller is not

60.4.2.6.11 ULPI VIEWPORT (optional)

The register provides indirect access to the ULPI PHY register set. Although the core performs access to the ULPI PHY register set, there may be extraordinary circumstances where software may need direct access.

CAUTION

Writes to the ULPI through the VIEWPORT can substantially harm standard USB operations. currently no usage model has been defined where software should need to execute writes directly to the ULPI.

NOTES

Executing read operations though the ULPI Viewport should have no harmful side effects to standard USB operations.

The ULPI Viewport is only synthesized in the design if the ULPI option has been purchased and installed, and the constant `VUSB_HS_PHY_ULPI` is set to `1`. If the ULPI interface is not enabled, this register always reads zeros.

There are two operations that can be performed with the ULPI Viewport, wakeup and read /write operations. The wakeup operation is used to put the ULPI interface into normal operation mode and re-enable the clock if necessary. A wakeup operation is required before accessing the registers when the ULPI interface is operating in low power mode, serial mode, or carkit mode. The ULPI state can be determined by reading the sync. state bit (ULPISS). If this bit is a one, then ULPI interface is running in normal operation mode and can accept read/write operations. If the ULPISS indicates a ‘0’ then read/write operations will not be able execute. Undefined behavior will result if ULPISS = 0 and a read or write operation is performed. To execute a wakeup operation, write all 32-bits of the ULPI Viewport where ULPIPORT is constructed appropriately and the ULPIWU bit is a ‘1’ and ULPIRUN bit is a ‘0’. Poll the ULPI Viewport until ULPIWU is zero for the operation to complete.

To execute a read or write operation, write all 32-bits of the ULPI Viewport where ULPIDATWR, ULPIADDR, ULPIPORT, ULPIRW are constructed appropriately and the ULPIRUN bit is a ‘1’. Poll the ULPI Viewport until ULPIRUN is zero for the operation to complete. Once ULPIRUN is zero, the ULPIDATRD will be valid if the operation was a read.

The polling method above could also be replaced and interrupt driven using the ULPI interrupt defined in the USBSTS and USBINTR registers. When a wakeup or read/write operation complete, the ULPI interrupt will be set.

Address: Base + 170h

Default Value: 00000000h

Attribute: Read/Write

Size: 32 bits

31	30	29	28	27	26 25 24	23 22 21 20 19 18 17 16	15 14 13 12 11 10 9 8	7 6 5 4 3 2 1 0
ULPIWU	ULPIRUN	ULPIRW	R	ULPISS	ULPIPORT	ULPIADDR	ULPIDATRD	ULPIDATWR

Figure 60-45. ULPI VIEWPORT

Table 60-51 shows the ULPI VIEWPORT field descriptions.

Table 60-51. ULPI VIEWPORT

Field	Description
ULPIWU	**ULPI Wakeup – Read/Write.** Writing the '1' to this bit will begin the wakeup operation. The bit will automatically transition to 0 after the wakeup is complete. Once this bit is set, the driver can not set it back to '0'. Note: The driver must never execute a wakeup and a read/write operation at the same time.
ULPIRUN	**ULPI Read/Write Run – Read/Write.** Writing the '1' to this bit will begin the read/write operation. The bit will automatically transition to 0 after the read/write is complete. Once this bit is set, the driver can not set it back to '0'. Note: The driver must never execute a wakeup and a read/write operation at the same time.
ULPIRW	**ULPI Read/Write Control – Read/Write.** (0) – Read; (1) – Write. This bit selects between running a read or write operation.
reserved	**Reserved.** This bit is reserved and its value has no effect on operation.
ULPISS	**ULPI Sync State – Read Only.** (1) – Normal Sync. State. (0) In another state (i.e. carkit, serial, low power) This bit represents the state of the ULPI interface. Before reading this bit, the ULPIPORT field should be set accordingly if used with the multi-port host. Otherwise, this field should always remain 0.
ULPIPORT	**ULPI Port Number – Read/Write.** For the wakeup or read/write operation to be executed, this value selects the port number the ULPI PHY is attached to in the multi-port host. The range is 0 to 7. This field should always be written as a 0 for the non-multi port products.
ULPIADDR	**ULPI Data Address – Read/Write.** When a read or write operation is commanded, the address of the operation is written to this field.
ULPIDATRD	**ULPI Data Read – Read Only.** After a read operation completes, the result is placed in this field.
ULPIDATWR	**ULPI Data Write – Read/Write**. When a write operation is commanded, the data to be sent is written to this field.

60.4.2.6.12 CONFIGFLAG

This register is not used in this implementation. A read from this register returns a constant of a 00000001h to indicate that all port routines default to this host controller.

Address: Base + 180h

Default Value: 00000001h

Attribute: Read Only

Size: 32 bits

60.4.2.6.13 PORTSCx

Address: Base + 184h + (4 × (Port Number – 1))

where Port Number = 1, 2, 3, …8

Default Value:

IIII00000000000000XX0000000000b (host mode)

IIII00000000000001XX0000000100b (device mode)

I	Implementation dependent
X	Unknown
Attribute:	RO, Read/Write, R/WC (field dependent)
Size:	32 bits

Host Controller

A host controller must implement one to eight port registers. The number of port registers implemented by a particular instantiation of a host controller is documented in the HCSPARAMs register. Software uses this information as an input parameter to determine how many ports need service.

This register is only reset when power is initially applied or in response to a controller reset. The initial conditions of a port are:

No device connected

Port disabled

If the port has port power control, this state remains until software applies power to the port by setting port power to one.

Device Controller

A device controller must implement only port register one and it does not support power control. Port control in device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling and allows software to put the PHY into low power suspend mode and disable the PHY clock.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PTS		STS	PTW	PSPD		R	PFSC	PHCD	WKOC	WKDS	WKCN	PTC[3:0]				PIC		PO	PP	LS		HSP	PR	SUSP	FPR	OCC	OCA	PEC	PE	CSC	CCS

Figure 60-46. PORTSCx—Port Status Control[1:8]

The register fields are described in Table 60-52.

Table 60-52. PORTSCx Field Descriptions

Field	Description
PTS	**Parallel Transceiver Select – Read/Write**. This register bit pair is used in conjunction with the configuration constant VUSB_HS_PHY_TYPE to control which parallel transceiver interface is selected. If VUSB_HS_PHY_TYPE is set for 0,1,2, or 3 then this bit is read only. If VUSB_HS_PHY_TYPE is 3,4,5, or 6 then this bit is read/write. This field is reset to: “00” if VUSB_HS_PHY_TYPE = 0,4 – UTMI/UTMI+ “01” if VUSB_HS_PHY_TYPE = 1,5 – Reserved “10” if VUSB_HS_PHY_TYPE = 2,6 – ULPI “11” if VUSB_HS_PHY_TYPE = 3,7 – Serial/1.1 PHY/IC USB(FS Only) This bit is not defined in the EHCI specification.

Table 60-52. PORTSCx Field Descriptions (continued)

STS	**Serial Transceiver Select – Read/Write**. This register bit is used in conjunction with the configuration constant VUSB_HS_PHY_SERIAL to control whether the parallel or serial transceiver interface is selected for FS and LS operation. The Serial Interface Engine can be used in combination with the UTMI+ or ULPI physical interface to provide FS/LS signaling instead of the parallel interface. If VUSB_HS_PHY_SERIAL is set for 0 or 1 then this bit is read only. If VUSB_HS_PHY_SERIAL is 3 or 4 then this bit is read/write. This bit has no effect unless Parallel Transceiver Select is set to UTMI+ or ULPI. The Serial/1.1 physical interface will use the Serial Interface Engine for FS/LS signaling regardless of this bit value. Note: This bit is reserved for future operation and is a placeholder adding dynamic use of the serial engine in accord with UMTI+ and ULPI characterization logic. This bit is not defined in the EHCI specification.
PTW	**Parallel Transceiver Width – Read/Write**. This register bit is used in conjunction with the configuration constant VUSB_HS_PHY8_16 to control whether the data bus width of the UTMI transceiver interface. If VUSB_HS_PHY8_16 is set for 0 or 1 then this bit is read only. If VUSB_HS_PHY8_16 is 2 or 3 then this bit is read/write. This bit is reset to 1 if VUSB_HS_PHY8_16 selects a default UTMI interface width of 16-bits else it is reset to 0. Writing this bit to 0 selects the 8-bit [60MHz] UTMI interface. Writing this bit to 1 selects the 16-bit [30MHz] UTMI interface. This bit has no effect if the Serial interfaces are selected. This bit is not defined in the EHCI specification.
PSPD	**Port Speed – Read Only.** This register field indicates the speed at which the port is operating. For HS mode operation in the host controller and HS/FS operation in the device controller the port routing steers data to the Protocol engine. For FS and LS mode operation in the host controller, the port routing steers data to the Protocol Engine w/ Embedded Transaction Translator. 00 Full Speed 01 Low Speed 10 High Speed This bit is not defined in the EHCI specification.
PFSC	**Port Force Full Speed Connect – Read/Write.** Default = 0b. Writing this bit to a 1b will force the port to only connect at Full Speed. It disables the chirp sequence that allows the port to identify itself as High Speed. This is useful for testing FS configurations with a HS host, hub or device. This bit is not defined in the EHCI specification. This bit is for debugging purposes.
PHCD	**PHY Low Power Suspend - Clock Disable (PLPSCD) – Read/Write.** Default = 0b. Writing this bit to a 1b will disable the PHY clock. Writing a 0b enables it. Reading this bit will indicate the status of the PHY clock. NOTE: The PHY clock cannot be disabled if it is being used as the system clock. In device mode, The PHY can be put into Low Power Suspend – Clock Disable when the device is not running (USBCMD Run/Stop=0b) or the host has signaled suspend (PORTSC SUSPEND=1b). Low power suspend will be cleared automatically when the host has signaled resume if using a circuit similar to that in. Before forcing a resume from the device, the device controller driver must clear this bit. In host mode, the PHY can be put into Low Power Suspend – Clock Disable when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software. See for more discussion on clock disable and power down issues. This bit is not defined in the EHCI specification.
WKOC	**Wake on Over-current Enable (WKOC_E) — Read/Write.** Default = 0b. Writing this bit to a one enables the port to be sensitive to over-current conditions as wake-up events. This field is zero if *Port Power(PP)* is zero. This bit is output from the controller as signal **pwrctl_wake_ovrcurr_en** (OTG/host core only) for use by an external power control circuit.

Table 60-52. PORTSCx Field Descriptions (continued)

WKDC	**Wake on Disconnect Enable (WKDSCNNT_E) — Read/Write.** Default=0b. Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events. This field is zero if *Port Power(PP)* is zero or in device mode. This bit is output from the controller as signal **pwrctl_wake_dscnnt_en** (OTG/host core only) for use by an external power control circuit.
WKCN	**Wake on Connect Enable (WKCNNT_E) — Read/Write.** Default=0b. Writing this bit to a one enables the port to be sensitive to device connects as wake-up events. This field is zero if *Port Power(PP)* is zero or in device mode. This bit is output from the controller as signal **pwrctl_wake_dscnnt_en** (OTG/host core only) for use by an external power control circuit.
PTC[3:0]	**Port Test Control — Read/Write.** Default = 0000b. Any other value than zero indicates that the port is operating in test mode. **ValueSpecific Test** 0000b TEST_MODE_DISABLE 0001b J_ STATE 0010b K_STATE 0011b SE0 (host) / NAK (device) 0100b Packet 0101b FORCE_ENABLE_HS 0110b FORCE_ENABLE_FS 0111b FORCE_ENABLE_LS 1000b to 1111bReserved Refer to Chapter 7 of the USB Specification Revision 2.0 Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org for details on each test mode. The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point. *Note: Low speed operations are not supported as a peripheral device.*
PIC[1:0]	**Port Indicator Control — Read/Write.** Default = Ob. Writing to this field has no effect if the P_INDICATOR bit in the HCSPARAMS register is a zero. If P_INDICATOR bit is a one, then the bit is: Bit ValueMeaning 00b Port indicators are off 01b Amber 10b Green 11b Undefined Refer to the USB Specification Revision 2.0 Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org for a description on how these bits are to be used. This field is output from the controller as signals **port_ind_ctl_1 & port_ind_ctl_0** for use by an external led driving circuit.
PO	**Port Owner—Read/Write**. Default = 0. This bit unconditionally goes to a 0 when the configured bit in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that an internal companion controller owns and controls the port. Port owner handoff is not implemented in this design, therefore this bit will always be 0.

Table 60-52. PORTSCx Field Descriptions (continued)

PP	**Port Power (PP)—Read/Write or Read Only.** The function of this bit depends on the value of the Port Power Switching (PPC) field in the HCSPARAMS register. The behavior is as follows: PPC PP Operation 0b 0b Read Only— A device controller with no OTG capability does not have port power control switches. 1b 1b/0b – RW. Host/OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (i.e. PP equals a 0), the port is non-functional and will not report attaches, detaches, etc. When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitional by the host controller driver from a one to a zero (removing power from the port). This feature is implemented in the host/OTG controller (PPC = 1). In a device only implementation port power control is not necessary, thus PPC and PP = 0.
LS[1:0]	**Line Status—Read Only.** These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. The encoding of the bits are: Bits [11:10]Meaning 00b SE0 10b J-state 01b K-state 11b Undefined In host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS. In device mode, the use of linestate by the device controller driver is not necessary.
HSP	**High-Speed Port — Read Only.** Default = 0b. When the bit is one, the host/device connected to the port is in high-speed mode and if set to zero, the host/device connected to the port is not in a high-speed mode. Note: HSP is redundant with PSPD(27:26) but will remain in the design for compatibility. This bit is not defined in the EHCI specification.
PR	**Port Reset** This field is zero if *Port Power(PP)* is zero. **In Host Mode:** Read/Write. 1=Port is in Reset. 0=Port is not in Reset. Default 0. When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. *This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver.* **In Device Mode:** This bit is a read only status bit. Device reset from the USB bus is also indicated in the USBSTS register.

Table 60-52. PORTSCx Field Descriptions (continued)

SUSP	Suspend **In Host Mode: Read/Write.** 1=Port in suspend state. 0=Port not in suspend state. Default=0. Port Enabled Bit and Suspend bit of this register define the port states as follows: Bits [Port Enabled, Suspend]Port State 0x Disable 10 Enable 11 Suspend When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB. The host controller will unconditionally set this bit to zero when software sets the *Force Port Resume* bit to zero. The host controller ignores a write of zero to this bit. If host software sets this bit to a one when the port is not enabled (i.e. *Port enabled* bit is a zero) the results are undefined. This field is zero if *Port Power(PP)* is zero in host mode. **In Device Mode: Read Only.** 1=Port in suspend state. 0=Port not in suspend state. Default=0. In device mode this bit is a read only status bit.
FPR	**Force Port Resume —Read/Write.** 1= Resume detected/driven on port. 0=No resume (K-state) detected/driven on port. Default = 0. **In Host Mode:** Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the *Port Change Detect* bit in the USBSTS register is also set to one. *This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver.* Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no affect because the port controller will time the resume operation clear the bit the port control state switches to HS or FS idle. This field is zero if *Port Power(PP)* is zero in host mode. This bit is not-EHCI compatible. **In Device mode:** After the device has been in Suspend State for 5ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. Also, when this bit transitions to a one because a J-to-K transition detected, the *Port Change Detect* bit in the USBSTS register is also set to one.
OCC	**Overcurrent Change—R/WC**. Default=0. 1=This bit gets set to one when there is a change to Over-current Active. Software clears this bit by writing a one to this bit position. For host/OTG implementations the user can provide over-current detection to the vbus_pwr_fault input for this condition. For device-only implementations this bit shall always be 0.
OCA	**Overcurrent Active—Read Only.** Default 0. 1=This port currently has an over-current condition. 0=This port does not have an over-current condition. This bit will automatically transition from one to zero when the over current condition is removed. For host/OTG implementations the user can provide over-current detection to the vbus_pwr_fault input for this condition. For device-only implementations this bit shall always be 0.

Table 60-52. PORTSCx Field Descriptions (continued)

PEC	**Port Enable/Disable Change—R/WC.** 1=Port enabled/disabled status has changed. 0=No change. Default = 0. **In Host Mode:** For the root hub, this bit gets set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this by writing a one to it. This field is zero if *Port Power(PP)* is zero. **In Device mode:** The device port is always enabled. (This bit will be zero)
PE	**Port Enabled/Disabled—Read/Write.** 1=Enable. 0=Disable. Default 0. In Host Mode: Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled, (0b) downstream propagation of data is blocked except for reset. This field is zero if *Port Power(PP)* is zero in host mode. **In Device Mode:** The device port is always enabled. (This bit will be one)
CSC	**Connect Status Change—R/WC.** 1 =Change in Current Connect Status. 0=No change. Default 0. **In Host Mode:** Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (i.e., the bit will remain set). Software clears this bit by writing a one to it. This field is zero if *Port Power(PP)* is zero in host mode. In Device Mode: This bit is undefined in device controller mode.
CCS	**Current Connect Status—Read Only**. **In Host Mode:** 1=Device is present on port. 0=No device is present. Default = 0. This value reflects the current state of the port, and may not correspond directly to the event that caused the *Connect Status Change* bit (Bit 1) to be set. This field is zero if *Port Power(PP)* is zero in host mode. **In Device Mode:** 1=Attached. 0=Not Attached. Default=0. A one indicates that the device successfully attached and is operating in either high speed or full speed as indicated by the High Speed Port bit in this register. A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.

60.4.2.6.14 OTGSC

Address: Base + 1A4h

Default Value: 00000020h

Attribute: RO, Read/Write, R/WC (field dependent)

Size: 32 bits

Host Controller

A host controller implements one On-The-Go (OTG) Status and Control register corresponding to Port 0 of the host controller.

The OTGSC register has four sections

- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 Msec time constant. Values on the status inputs that do not persist for more than 1 Msec will not cause an update of the status input register, or cause an OTG interrupt.

See also USBMODE register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPIE	1msE	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	R	DPIS	1mss	BSEIS	BSVIS	ASVIS	AVVIS	IDIS	R	DPS	1msT	BSE	BSV	ASV	AVV	ID	R		IDPU	DP	OT	R	VC	VD

Figure 60-47. OTGSC—OTG Status Control

The register fields are described in Table 60-53.

Table 60-53. OTGSC Field Descriptions

Field	Description
DPIE	Data Pulse Interrupt Enable
1msE	1 millisecond timer Interrupt Enable – Read/Write
BSEIE	B Session End Interrupt Enable – Read/Write. Setting this bit enables the B session end interrupt.
BSVIE	B Session Valid Interrupt Enable – Read/Write. Setting this bit enables the B session valid interrupt.
ASVIE	A Session Valid Interrupt Enable – Read/Write. Setting this bit enables the A session valid interrupt.
AVVIE	A VBus Valid Interrupt Enable – Read/Write. Setting this bit enables the A VBus valid interrupt.
IDIE	USB ID Interrupt Enable – Read/Write. Setting this bit enables the USB ID interrupt.

Table 60-53. OTGSC Field Descriptions (continued)

DPIS	Data Pulse Interrupt Status – Read/Write to Clear. This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE.CM = Host (11) and PORTSC(0).PortPower = Off (0). Software must write a one to clear this bit.
1msS	1 millisecond timer Interrupt Status – Read/Write to Clear. This bit is set once every millisecond. Software must write a one to clear this bit.
BSEIS	B Session End Interrupt Status – Read/Write to Clear. This bit is set when VBus has fallen below the B session end threshold. Software must write a one to clear this bit
BSVIS	**B Session Valid Interrupt Status – Read/Write to Clear**. This bit is set when VBus has either risen above or fallen below the B session valid threshold (0.8 VDC). Software must write a one to clear this bit.
ASVIS	**A Session Valid Interrupt Status – Read/Write to Clear**. This bit is set when VBus has either risen above or fallen below the A session valid threshold (0.8 VDC). Software must write a one to clear this bit.
AVVIS	**A VBus Valid Interrupt Status – Read/Write to Clear**. This bit is set when VBus has either risen above or fallen below the VBus valid threshold (4.4 VDC) on an A device. Software must write a one to clear this bit.
IDIS	USB ID Interrupt Status – Read/Write. This bit is set when a change on the ID input has been detected. Software must write a one to clear this bit.
DPS	Data Bus Pulsing Status – Read Only. A '1' indicates data bus pulsing is being detected on the port.
1msT	1 millisecond timer toggle - Read Only. This bit toggles once per millisecond.
BSE	B Session End – Read Only. Indicates VBus is below the B session end threshold.
BSV	B Session Valid – Read Only. Indicates VBus is above the B session valid threshold.
ASV	A Session Valid – Read Only. Indicates VBus is above the A session valid threshold.
AVV	A VBus Valid – Read Only. Indicates VBus is above the A VBus valid threshold.
ID	USB ID – Read Only. 0 = A device, 1 = B device
IDPU	ID Pullup – Read/Write This bit provide control over the ID pull-up resister; 0 = off, 1 = on [default]. When this bit is 0, the ID input will not be sampled.
DP	Data Pulsing – Read/Write. Setting this bit causes the pullup on DP to be asserted for data pulsing during SRP.
OT	OTG Termination – Read/Write. This bit must be set when the OTG device is in device mode, this controls the pulldown on DM.

Table 60-53. OTGSC Field Descriptions (continued)

VC	VBUS Charge – Read/Write. Setting this bit causes the VBus line to be charged. This is used for VBus pulsing during SRP.
VD	VBUS_Discharge – Read/Write. Setting this bit causes VBus to discharge through a resistor.

60.4.2.6.15 USBMODE

Address: Base + 1A8h

Default Value: 00000000h (otg implementation – mode not selected)
00000003h (host mode)
00000002h (device mode))

Attribute: R/WO, Read Only

Size: 32 bits

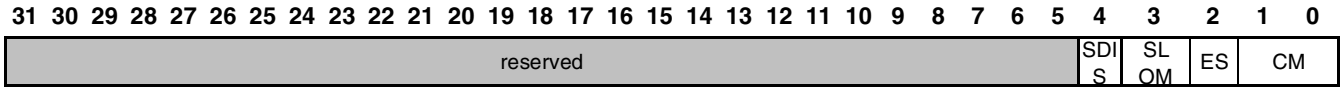


Figure 60-48. USBMODE—USB Device Mode

The register fields are described in Table 60-54.

Table 60-54. USBMODE Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
SDIS	Stream Disable Mode. (0 – Inactive [default]; 1 – Active) **Device Mode:** Setting to a '1' disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received will be responded to with a NYET handshake when stream disable is active. **Host Mode:** Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. Note: Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING and TXTTFILLTUNING [MPH Only] to characterize the adjustments needed for the scheduler when using this feature. Note: The use of this feature substantially limits of the overall USB performance that can be achieved.
SLOM	**Setup Lockout Mode.** In device mode, this bit controls behavior of the setup lock mechanism. See Control Endpoint Operation Model. 0 – Setup Lockouts On (default); 1 – Setup Lockouts Off (DCD requires use of Setup Data Buffer Tripwire in USBCMD)

Table 60-54. USBMODE Field Descriptions (continued)

ES	**Endian Select – Read/Write.** This bit can change the byte alignment of the transfer buffers to match the host microprocessor. The bit fields in the microprocessor interface and the data structures are unaffected by the value of this bit because they are based upon the 32-bit word. **Bit** **Meaning** 0 Little Endian [Default] 1 Big Endian
CM[1:0]	**Controller Mode – R/WO.** Controller mode is defaulted to the proper mode for host only and device only implementations. For those designs that contain both host & device capability, the controller will default to an idle state and will need to be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the *RESET* bit in the USBCMD register before reprogramming this register. **Bit** **Meaning** 00 Idle [Default for combination host/device] 01 Reserved 10 Device Controller [Default for device only controller] 11 Host Controller [Default for host only controller]

60.4.2.6.16 ENDPTSETUPSTAT

Address: Base + 1ACh

Default Value: 00000000h

Attribute: R/WC

Size: 32 bits

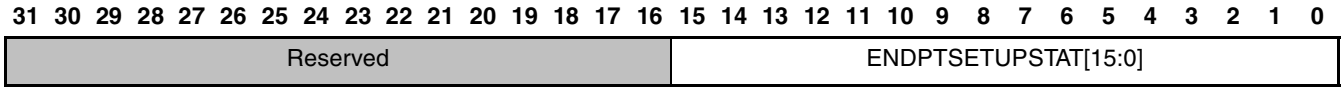


Figure 60-49. ENDPTSETUPSTAT—Endpoint Setup Status

The register fields are described in Table 60-55.

Table 60-55. ENDPTSETUPSTAT Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
ENDPTSETUPSTAT [15:0]	**Setup Endpoint Status.** For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lock our mechanism is engaged. See Managing Endpoints in the Device Operational Model. This register is only used in device mode.

60.4.2.6.17 ENDPTPRIME

Address: Base + 1B0h

Default Value: 00000000h

Attribute: R/WS

Size: 32 bits

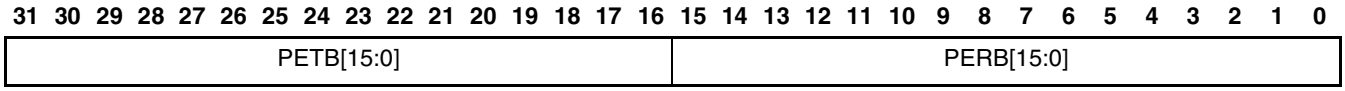


Figure 60-50. ENDPTPRIME—Endpoint Initialization

This register is only used in device mode. The register fields are described in Table 60-56.

Table 60-56. ENDPTPRIME Field Descriptions

Field	Description
PETB [15:0]	**Prime Endpoint Transmit Buffer – R/WS.** For each endpoint a corresponding bit is used to request that a buffer prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. Note: These bits will be momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PETB[15] – Endpoint #15 PETB[1] – Endpoint #1 PETB[0] – Endpoint #0
PERB [15:0]	**Prime Endpoint Receive Buffer – R/WS.** For each endpoint, a corresponding bit is used to request a buffer prepare for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint. Hardware will automatically use this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware will clear this bit when the associated endpoint(s) is (are) successfully primed. Note: These bits will be momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. Bit 15 – Endpoint #15 Bit 1 – Endpoint #1 Bit 0 – Endpoint #0

60.4.2.6.18 ENDPTFLUSH

Address: Base + 1B4h

Default Value: 00000000h

Attribute: R/WS

Size: 32 bits

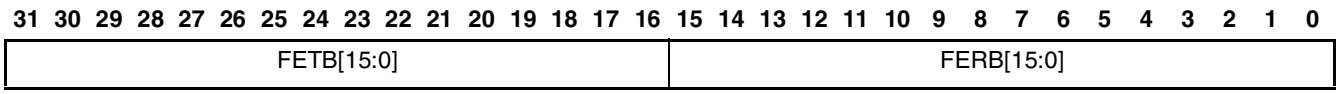


Figure 60-51. ENDPTFLUSH—Endpoint De-Initialize

This register is only used in device mode. The register fields are described in Table 60-57.

Table 60-57. ENDPTFLUSH Field Descriptions

Field	Description
FETB [15:0]	**Flush Endpoint Transmit Buffer – R/WS.** Writing a one to a bit(s) in this register will cause the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer will continue until completion. Hardware will clear this register after the endpoint flush operation is successful. FETB[15] – Endpoint #15 FETB[1] – Endpoint #1 FETB[0] – Endpoint #0
FERB [15:0]	**Flush Endpoint Receive Buffer – R/WS.** Writing a one to a bit(s) will cause the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer will continue until completion. Hardware will clear this register after the endpoint flush operation is successful. Bit 15 – Endpoint #15 Bit 1 – Endpoint #1 Bit 0 – Endpoint #0

60.4.2.6.19 ENDPTSTAT

Address: Base + 1B8h

Default Value: 00000000h

Attribute: Read Only

Size: 32 bits

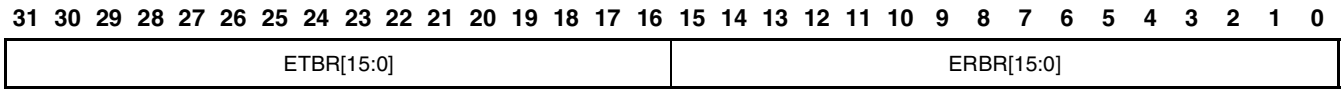


Figure 60-52. ENDPTSTAT—Endpoint Status

This register is only used in device mode. The register fields are described in Table 60-57.

Table 60-58. ENDPTSTAT Field Descriptions

Field	Description
ETBR [15:0]	**Endpoint Transmit Buffer Ready—Read Only.** One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There will always be a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. Note: These bits will be momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ETBR[15]– Endpoint #15 ETBR[1]– Endpoint #1 ETBR[0]– Endpoint #0
ERBR [15:0]	**Endpoint Receive Buffer Ready—Read Only.** One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There will always be a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register. Note: These bits will be momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ERBR[15]– Endpoint #15 ERBR[1]– Endpoint #1 ERBR[0]– Endpoint #0

60.4.2.6.20 ENDPTCOMPLETE

Address: Base + 1BCh

Default Value: 00000000h

Attribute: R/WC

Size: 32 bits

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETCE[15:0]																ERCE[15:0]															

Figure 60-53. ENDPTCOMPLETE—Endpoint Compete

This register is only used in device mode. The register fields are described in Table 60-57.

Table 60-59. ENDPTCOMPLETE Field Descriptions

Field	Description
ETCE [15:0]	**Endpoint Transmit Complete Event – R/WC.** Each bit indicates a transmit event (IN/INTERRUPT) occurred and software should read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit will be set simultaneously with the *USBINT*. Writing a one will clear the corresponding bit in this register. ETCE[15]– Endpoint #15 ETCE[1]– Endpoint #1 ETCE[0]– Endpoint #0
ERCE [15:0]	**Endpoint Receive Complete Event – RW/C.** Each bit indicates a received event (OUT/SETUP) occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit will be set simultaneously with the *USBINT*. Writing a one will clear the corresponding bit in this register. ERCE[15]– Endpoint #15 ERCE[1]– Endpoint #1 ERCE[0]– Endpoint #0

60.4.2.6.21 ENDPTCTRL0

Address: Base + 1C0h

Default Value: 0080008h

Attribute: Read Only, Read/Write, R/WC (field dependent)

Size: 32 bits

Every device implements Endpoint0 as a control endpoint.

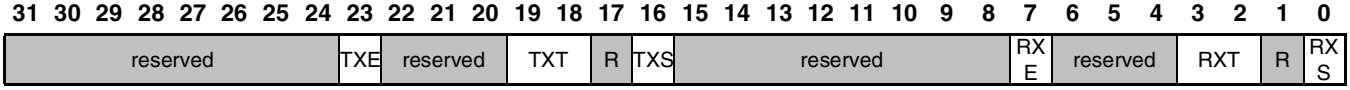


Figure 60-54. ENDPTCTRL0—Endpoint Control 0

The register fields are described in the following table.

Table 60-60. ENDPTCTRL0 Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.
TXE	**TX Endpoint Enable** 1 – Enabled Endpoint0 is always enabled.
TXT[1:0]	**TX Endpoint Type – Read/Write** 00 – Control Endpoint0 is fixed as a Control End Point.
R	**Reserved.** Bit reserved and should be set to zero.

Table 60-60. ENDPTCTRL0 Field Descriptions (continued)

TXS	**TX Endpoint Stall – Read/Write** 0 – End Point OK [Default] 1 – End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software or it will automatically be cleared upon receipt of a new SETUP request.
RXE	**RX Endpoint Enable** 1 – Enabled Endpoint0 is always enabled.
RXT[1:0]	**RX Endpoint Type – Read/Write** 00 – Control Endpoint0 is fixed as a Control End Point.
RXS	**RX Endpoint Stall – Read/Write** 0 – End Point OK. [Default] 1 – End Point Stalled Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue returning STALL until the bit is cleared by software or it will automatically be cleared upon receipt of a new SETUP request.

60.4.2.6.22 ENDPTCTRL1 – ENDPTCTRL15

There is an ENDPTCTRLx register for each endpoint in a device.

Address: Base + 1C0h+(4*(EndPoint Number))

Default Value: 00000000h

Attribute: Read/Write

Size: 32 bits

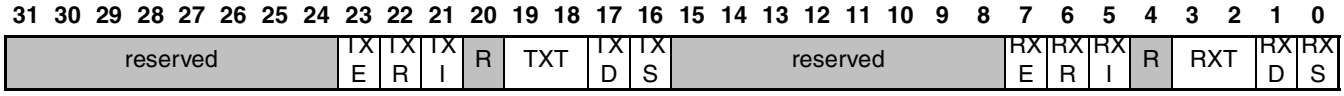


Figure 60-55. ENDPTCTRL1–15—Endpoint Control 1 to 15

CAUTION

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (IE. Bulk-type). leaving an unconfigured endpoint control will cause undefined behavior for the data pid tracking on the active endpoint/direction.

The register fields are described in Table 60-61.

Table 60-61. Endpoint Control*x* Field Descriptions

Field	Description
Reserved	**Reserved.** These bits are reserved and should be set to zero.

Table 60-61. Endpoint Control*x* Field Descriptions *(continued)*

TXE	**TX Endpoint Enable** 0 – Disabled [Default] 1 – Enabled An Endpoint should be enabled only after it has been configured.
TXR	**TX Data Toggle Reset (WS)** Write 1 – Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
TXI	**TX Data Toggle Inhibit** 0 – PID Sequencing Enabled. [Default] 1 – PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
R	**Reserved.** Bit reserved and should be set to zero.
TXT[1:0]	**TX Endpoint Type – Read/Write** 00 – Control 01 – Isochronous 10 – Bulk 11 – Interrupt
TXD	**TX Endpoint Data Source – Read/Write** 0 – Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
TXS	**TX Endpoint Stall – Read/Write** 0 – End Point OK 1 – End Point Stalled This bit will be set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It will be cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue to returning STALL until this bit is either cleared by software or automatically cleared as above.
RXE	**RX Endpoint Enable** 0 – Disabled [Default] 1 – Enabled An Endpoint should be enabled only after it has been configured.
RXR	**RX Data Toggle Reset (WS)** Write 1 – Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
RXI	**RX Data Toggle Inhibit** 0 – Disabled [Default] 1 – Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit will cause this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
RXT[1:0]	**RX Endpoint Type – Read/Write** 00 – Control 01 – Isochronous 10 – Bulk 11 – Reserved

Table 60-61. Endpoint Control*x* Field Descriptions *(continued)*

RXD	**RX Endpoint Data Sink – Read/Write – TBD** 0 – Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
RXS	**RX Endpoint Stall – Read/Write** 0 – End Point OK. [Default] 1 – End Point Stalled This bit will be set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It will be cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint, Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It will continue to returning STALL until this bit is either cleared by software or automatically cleared as above,

60.4.2.7 OTG Operations

This section discusses the following:

- Section 60.4.2.7.1, Register Bits”
- Section 60.4.2.7.2, Hardware Assist”

60.4.2.7.1 Register Bits

In the previous section, the register interface has behaviors described for device mode and behaviors described for host mode. However, during OTG operations it is necessary to perform tasks independent of the controller mode.

Note also from Section 60.4.2.6.22, ENDPTCTRL1 – ENDPTCTRL15,” that the only way to transition the controller mode out of host or device mode is with the controller reset bit. Therefore, it is also necessary for the OTG tasks to be performed independent of a controller reset as well as independent of the controller mode.

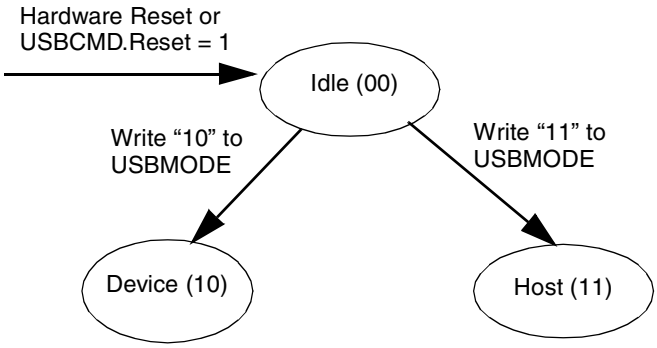


Figure 60-56. Controller Mode

To this end, the following list shows the register bits that are used for OTG operations, which are independent of the controller mode. They are also not affected by a write to the reset bit in the USBCMD register:

- All Identification Registers

- All Device/Host Capability Registers
- OTGSC—All bits
- PORTSC
  - Physical Interface Select
  - Physical Interface Serial Select
  - Physical Interface Data Width
  - Physical Interface Low Power
  - Physical Interface Wake Signals
  - Port Indicators
  - Port Power

60.4.2.7.2 Hardware Assist

The hardware assist provides automated response and sequencing that may not be possible to software with significant interrupt latency response times. The use of this additional circuitry is optional and can be used to assist the three sequences below.

Auto-Reset

When the HAAR is set to one, the host will automatically start a reset after a connect event. This shortcuts the normal process where software is notified of the connect event and starts the reset. Software will still receive notification of the connect event but should not write the reset bit when the HAAR is set. Software will be notified again after the reset is complete via the enable change bit in the PORTSC register which cause a port change interrupt.

This assist will ensure the OTG parameter TB_ACON_BSE0_MAX = 1ms is met.

Data-Pulse

Writing a one to HADP will start a data pulse of approximately 7ms in duration and then automatically cease the data pulsing. During the data pulse, the DP will be set and then cleared. This automation relieves software from accurately controlling the data-pulse duration. During the data pulse, the HCD can poll to see that the HADP and DP bit have returned low to recognize the completion or simply launch the data pulse and wait to see if a VBUS Valid interrupt occurs when the A-side supplies bus power.

This assist will ensure data pulsing meets the OTG requirement of > 5ms and < 10ms.

B-Disconnect to A-Connect

During HNP, the B-disconnect occurs from the OTG A_suspend state and within 3 ms, the A device must enable the pullup on the DP leg in the A-peripheral state. When HABA is set, the Host Controller port is in suspend mode, and the device disconnects, then this hardware assist begins.

1. Reset the OTG core.
2. Write the OTG core into device mode.
3. Write the device run bit to a 1 and enable necessary interrupts including:

- USB Reset Enable (URE) ; enables interrupt on usb bus reset to device
- Sleep Enable (SLE) ; enables interrupt on device suspend
- Port Change Detect Enable (PCE) ; enables interrupt on device connect

When software has enabled this hardware assist, it must not interfere during the transition and should not write any register in the core until it gets an interrupt from the device controller signifying that a reset interrupt has occurred or at least first verify that the core has entered device mode. HCD/DCD must not activate the core soft reset at any time since this action is performed by hardware. During the transition, the software may see an interrupt from the disconnect and/or other spurious interrupts (ie. SOF/etc.) that may or may not cascade and my be cleared by the soft reset depending on the software response time.

After the core has entered device mode by the hardware assist, the DCD must ensure that the ENDPTLISTADDR is programmed properly before the host sends a setup packet. Since the end of the reset duration, which may be initiated quickly (a few microseconds) after connect, will require at a minimum 50 ms, this is the time for which the DCD must be ready to accept setup packets after having received notification that the reset has been detected or simply that the OTG is in device mode which ever occurs first.

In the case where the A-peripheral fails to see a reset after the controller enters device mode and engages the DP-pullup, the device controller interrupt the DCD signifying that a suspend has occurred.

This assist will ensure the parameter TA_BDIS_ACON_MAX = 3ms is met.

60.4.3 Host Data Structures

This section defines the interface data structures used to communicate control, status, and data between HCD (software) and the Enhanced Host Controller (hardware). The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of a Periodic Schedule, Periodic Frame List, Asynchronous Schedule, Isochronous Transaction Descriptors, Split-transaction Isochronous Transfer Descriptors, Queue Heads, and Queue Element Transfer Descriptors.

The periodic frame list is the root of all periodic (isochronous and interrupt transfer type) support for the host controller interface. The asynchronous list is the root for all the bulk and control transfer type support. Isochronous data streams are managed using Isochronous Transaction Descriptors. Isochronous split-transaction data streams are managed with Split-transaction Isochronous Transfer Descriptors. All Interrupt, Control, and Bulk data streams are managed via queue heads and Queue Element Transfer Descriptors. These data structures are optimized to reduce the total memory footprint of the schedule and to reduce (on average) the number of memory accesses needed to execute a USB transaction.

Note that software must ensure that no interface data structure reachable by the EHCI host controller spans a 4K-page boundary.

The data structures defined in this section are (from the host controller’s perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

60.4.3.1 Periodic Frame List

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the *PERIODICLISTBASE* address register and the *FRINDEX* register. The periodic schedule is based on an array of pointers called the Periodic Frame List. The *PERIODICLISTBASE* address register is combined with the *FRINDEX* register to produce a memory pointer into the frame list. The Periodic Frame List implements a *sliding window* of work over time.

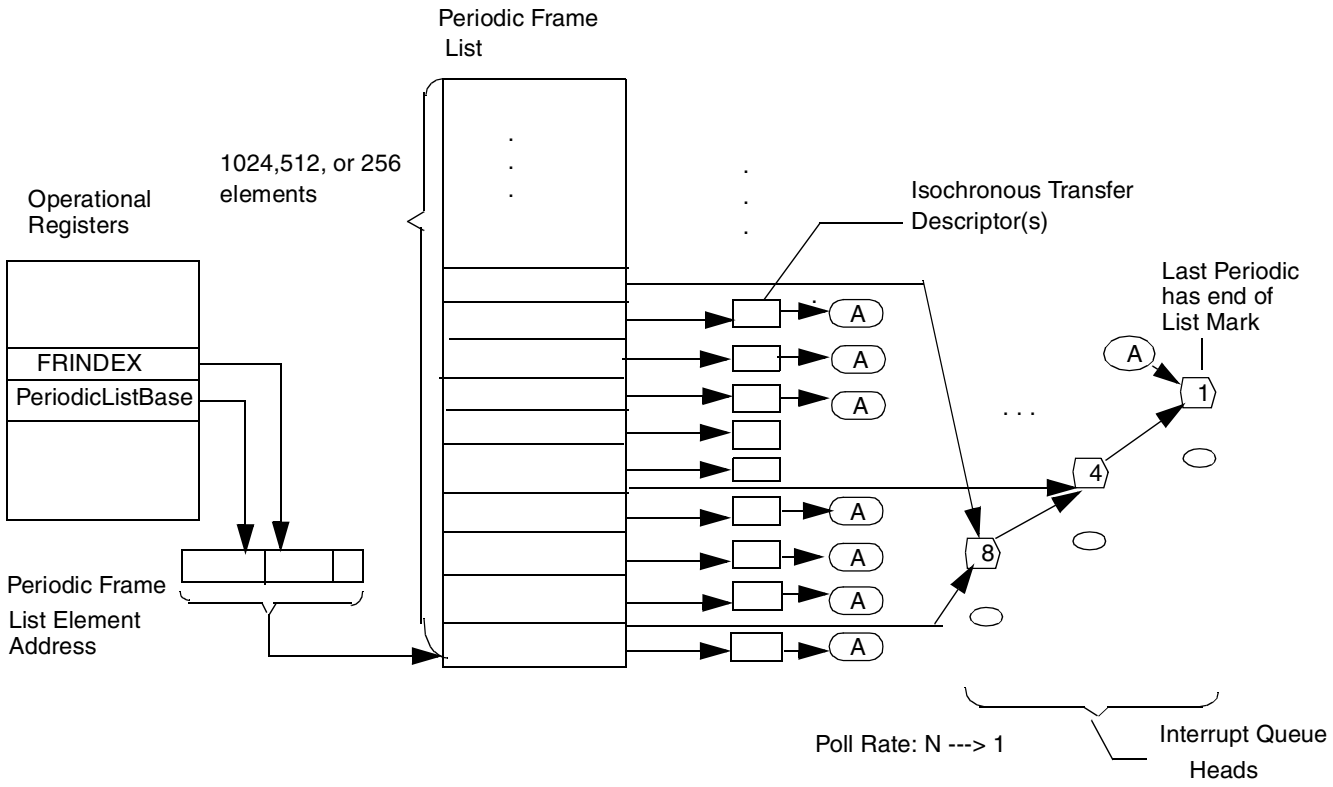


Figure 60-57. Periodic Schedule Organization

[1] Split transaction Interrupt, Bulk and Control are also managed using queue heads and queue element transfer descriptors.

The periodic frame list is a 4K-page aligned array of Frame List Link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to system software via the HCCPARAMS register. If non-programmable, the length is 1024 elements. If programmable, the length can be selected by system software as one of 256, 512, or 1024 elements. An implementation must support all three sizes. Programming the size (i.e. the number of elements) is accomplished by system software writing the appropriate value into *Frame List Size* field in the USBCMD register.

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current micro-frame. The link pointers are aligned on DWord boundaries within the Frame List.

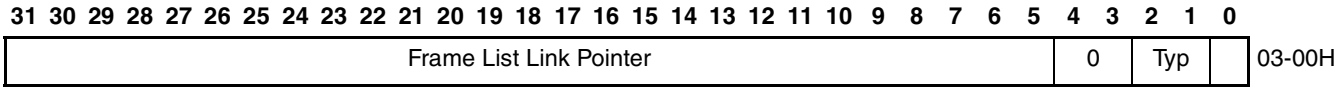


Figure 60-58. Format of Frame List Element Pointer

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced object may be an isochronous transfer descriptor for high-speed devices, a split-transaction isochronous transfer descriptor (for full-speed isochronous endpoints), or a queue head (used to support high-, full- and low-speed interrupt). System software should not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to key the host controller as to the type of object the pointer is referencing.

The least significant bit is the *T-Bit* (bit 0). When this bit is set to a one, the host controller will never use the value of the frame list pointer as a physical memory pointer. The *Typ* field is used to indicate the exact type of data structure being referenced by this pointer. The value encodings are shown in Table 60-62.

Table 60-62. Typ Field Value Definitions

Value	Meaning
00b	Isochronous Transfer Descriptor
01b	Queue Head
10b	Split Transaction Isochronous Transfer Descriptor.
11b	Frame Span Traversal Node.

60.4.3.2 Asynchronous List Queue Head Pointer

The Asychronous Transfer List (based at the ASYNCLISTADDR register) is where all the control and bulk transfers are managed. Host controllers use this list only when it reaches the end of the periodic list, the periodic list is disabled, or the periodic list is empty.

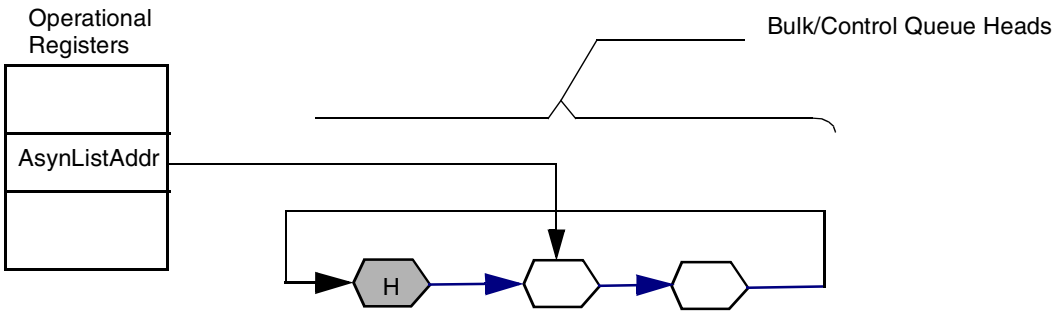
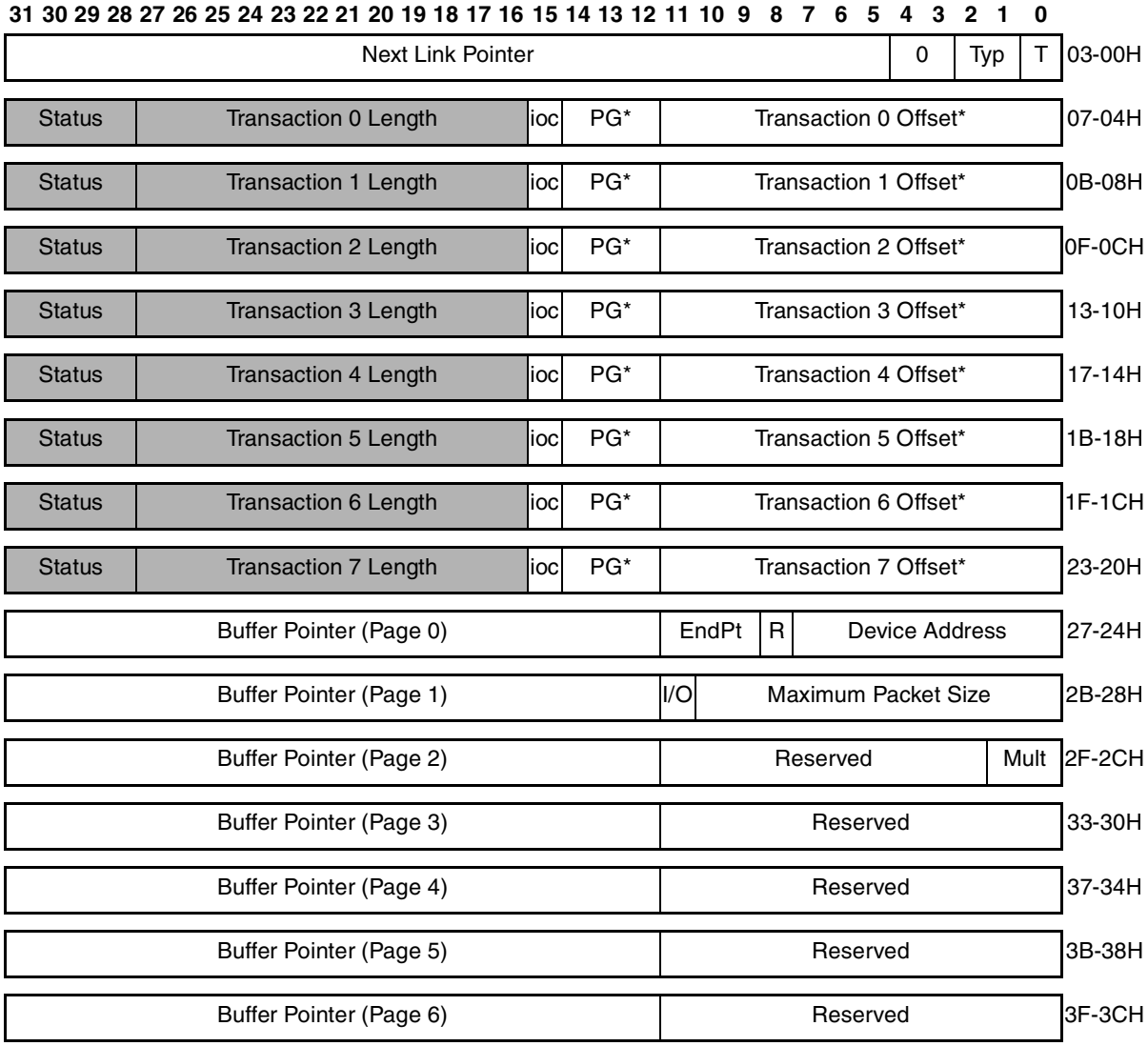


Figure 60-59. Asynchronous Schedule Organization

The Asynchronous list is a simple circular list of queue heads. The ASYNCLISTADDR register is simply pointer to the *next* queue head. This implements a pure round-robin service for all queue heads linked into the asynchronous list.

60.4.3.3 Isochronous (High-Speed) Transfer Descriptor (iTD)

The format of an isochronous transfer descriptor is illustrated in Figure 60-60. This structure is used only for high-speed isochronous endpoints. All other transfer types should use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.



Host Controller Read/Write Host Controller Read Only.

Note: These fields may be modified by the host controller if the I/O field indicates an OUT.

Figure 60-60. Isochronous Transaction Descriptor (iTD)

60.4.3.3.1 Next Link Pointer

The first DWord of an iTD is a pointer to the next schedule data structure, as shown in Table 60-63.

Table 60-63. Next Schedule Element Pointer

Bit	Description
31–5	**Link Pointer (LP).** These bits correspond to memory address signals [31:5], respectively. This field points to another Isochronous Transaction Descriptor (iTD/siTD) or Queue Head (QH).
4–3	**Reserved.** These bits are reserved and their value has no effect on operation. Software should initialize this field to zero.
2–1	**QH/(s)iTD Select (Typ).** This field indicates to the Host Controller whether the item referenced is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: ValueMeaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0	Terminate (T). 1= Link Pointer field is not valid. 0= Link Pointer field is valid.

60.4.3.3.2 iTD Transaction Status and Control List

DWords 1 through 8 are eight slots of transaction control and status. Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions).
- Buffer offset. The *PG* and *Transaction X Offset* fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list, to execute a transaction on the USB.

Table 60-64. iTD Transaction Status and Control Field Descriptions

Bit	Description
31–28	**Status.** This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding: 31 **Active.** Set to one by software to enable the execution of an isochronous transaction by the Host Controller. When the transaction associated with this descriptor is completed, the Host Controller sets this bit to zero indicating that a transaction for this element should not be executed when it is next encountered in the schedule. 30 **Data Buffer Error.** Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, no action is necessary 29 **Babble Detected.** Set to one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor. 28 **Transaction Error (XactErr).** Set to one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit may only be set for isochronous IN transactions.

Table 60-64. iTD Transaction Status and Control Field Descriptions (continued)

27–16	**Transaction X Length**. For an OUT, this field is the number of data bytes the host controller will send during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN, the initial value of the endpoint to deliver. During the status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (for example, 0‡zero length data, 1‡one byte, 2‡two bytes, etc.). The maximum value this field may contain is 0xC00 (3072).
15	**Interrupt On Complete (IOC).** If this bit is set to one, it specifies that when this transaction completes, the Host Controller should issue an interrupt at the next interrupt threshold.
14–12	**Page Select (PG).** These bits are set by software to indicate which of the buffer page pointers the offset field in this slot should be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.
11–0	**Transaction X Offset**. This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. This field is concatenated onto the buffer page pointer indicated in the adjacent *PG* field to produce the starting buffer address for this transaction.

60.4.3.3.3 iTD Buffer Page Pointer List (Plus)

DWords 9–-15 of an isochronous transaction descriptor are nominally page pointers (4K aligned) to the data buffer for this transfer descriptor. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous. Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) × 1024 (maximum packet size) × 8 (transaction records) (24576 bytes) to be moved with this data structure, regardless of the alignment offset of the first page.

Since each pointer is a 4K aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

Table 60-65. iTD Buffer Pointer Page 0 (Plus)

Bit	Description
31–12	**Buffer Pointer (Page 0).** This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11–8	**Endpoint Number (Endpt).** This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	**Reserved.** Bit reserved for future use and should be initialized by software to zero.
6–0	**Device Address.** This field selects the specific device serving as the data source or sink.

Table 60-66. iTD Buffer Pointer Page 1 (Plus)

Bit	Description
31–12	**Buffer Pointer (Page 1).** This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].

Table 60-66. iTD Buffer Pointer Page 1 (Plus) (continued)

11	**Direction (I/O). 0 = OUT; 1 = IN.** This field encodes whether the high-speed transaction should use an IN or OUT PID.
10–0	**Maximum Packet Size.** This directly corresponds to the maximum packet size of the associated endpoint (*wMaxPacketSize*). This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (for example, per micro-frame). This field is used with the *Multi* field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. Software should not set a value larger than 1024 (400h). Any value larger yields undefined results.

Table 60-67.
Table 60-68. iTD Buffer Pointer Page 2 (Plus)

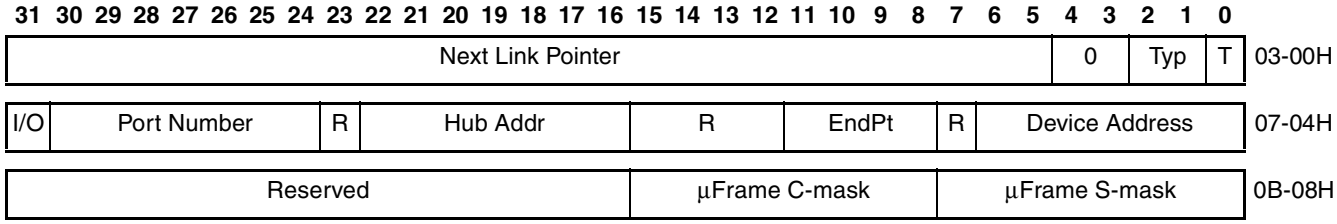
Bit	Description
31–12	**Buffer Pointer.** This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11–2	**Reserved.** This bit reserved for future use and should be set to zero.
1–0	**Multi.** This field is used to indicate to the host controller the number of transactions that should be executed per transaction description (for example, per micro-frame). The valid values are: ValueMeaning 00bReserved. A zero in this field yields undefined results. 01bOne transaction to be issued for this endpoint per micro-frame 10bTwo transactions to be issued for this endpoint per micro-frame 11bThree transactions to be issued for this endpoint per micro-frame

Table 60-69.
Table 60-70. iTD Buffer Pointer Page 3–6

Bit	Description
31–12	**Buffer Pointer.** This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11–0	**Reserved.** These bits reserved for future use and should be set to zero.

60.4.3.4 Split Transaction Isochronous Transfer Descriptor (siTD)

All full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.



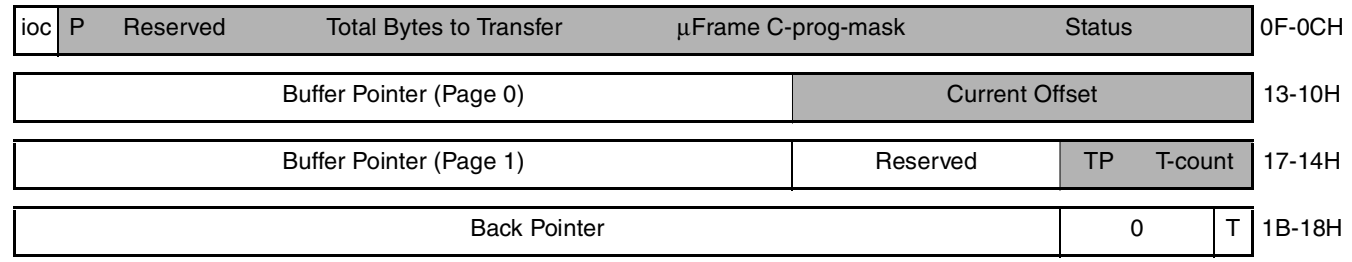


Figure 60-61. Split-transaction Isochronous Transaction Descriptor (siTD)

60.4.3.4.1 Next Link Pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

Table 60-71. Next Link Pointer

Bit	Description
31–5	**Next Link Pointer (LP).** This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4–3	**Reserved.** These bits must be written as zeros.
2–1	**QH/(s)iTD Select (Typ).** This field indicates to the Host Controller whether the item referenced is an iTD/siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: ValueMeaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor 11b FSTN (frame span traversal node)
0	**Terminate (T).** 1 = Link Pointer field is not valid. 0 = Link Pointer is valid.

60.4.3.4.2 siTD Endpoint Capabilities/Characteristics

DWords 1 and 2 specify static information about the full-speed endpoint, the addressing of the parent Companion Controller, and micro-frame scheduling control.

Table 60-72. Endpoint and Transaction Translator Characteristics

Bit	Description
31	**Direction (I/O).0 = OUT; 1 = IN.** This field encodes whether the full-speed transaction should be an IN or OUT.
30–24	**Port Number.** This field is the port number of the recipient Transaction Translator.
23	**Reserved.** Bit reserved and should be set to zero.
22–16	**Hub Address.** This field holds the device address of the Companion Controllers' hub.
15–12	**Reserved.** Field reserved and should be set to zero.
11–8	**Endpoint Number (Endpt).** This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.

Table 60-72. Endpoint and Transaction Translator Characteristics (continued)

7	**Reserved.** Bit is reserved for future use. It should be set to zero.
6–0	**Device Address.** This field selects the specific device serving as the data source or sink.

Table 60-73. Microframe Schedule Control

Bit	Description
31–16	**Reserved.** This field reserved for future use. It should be set to zero.
15–8	**Split Completion Mask (μFrame C-Mask).** This field (along with the *Active* and *SplitX- state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute complete-split transactions. When the criteria for using this field is met, an all zeros value has undefined behavior. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the μ*Frame C-Mask* field is a one, then this siTD is a candidate for transaction execution. There may be more than one bit in this mask set.
7–0	**Split Start Mask (μFrame S-mask).** This field (along with the *Active* and *SplitX-state* fields in the *Status* byte) is used to determine during which micro-frames the host controller should execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the μ*Frame S-mask* field is a one, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list has undefined results.

60.4.3.4.3 siTD Transfer State

DWords 3–6 are used to manage the state of the transfer.

Table 60-74. siTD Transfer Status and Control

<table>
<tr><th>Bit</th><th colspan="2">Description</th></tr>
<tr><td>31</td><td colspan="2">Interrupt On Complete (ioc). 0 = Do not interrupt when transaction is complete. 1 = Do interrupt when transaction is complete. When the host controller determines that the split transaction has completed it will assert a hardware interrupt at the next interrupt threshold.</td></tr>
<tr><td>30</td><td colspan="2">Page Select (P). Used to indicate which data page pointer should be concatenated with the CurrentOffset field to construct a data buffer pointer (0 selects Page 0 pointer and 1 selects Page 1). The host controller is not required to write this field back when the siTD is retired (Active bit transitioned from a one to a zero).</td></tr>
<tr><td>29–26</td><td colspan="2">Reserved. This field reserved for future use and should be set to zero.</td></tr>
<tr><td>25–16</td><td colspan="2">Total Bytes To Transfer. This field is initialized by software to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh)</td></tr>
<tr><td>15–8</td><td colspan="2">μFrame Complete-split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.</td></tr>
<tr><td>7–0</td><td colspan="2">Status. This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:</td></tr>
<tr><td></td><td>Bit</td><td>Definition</td></tr>
<tr><td></td><td>7</td><td>Active. Set to one by software to enable the execution of an isochronous split transaction by the Host Controller.</td></tr>
<tr><td></td><td>6</td><td>ERR. Set to a one by the Host Controller when an ERR response is received from the Companion Controller.</td></tr>
</table>

Table 60-74. siTD Transfer Status and Control

	5	**Data Buffer Error**. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the Host Controller will transmit an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
	4	**Babble Detected.** Set to a one by the Host Controller during status update when" babble" is detected during the transaction generated by this descriptor.
	3	**Transaction Error (XactErr).** Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit will only be set for IN transactions.
	2	**Missed Micro-Frame.** The host controller detected that a host-induced hold- off caused the host controller to miss a required complete-split transaction.
	1	**Split Transaction State (SplitXstate).** The bit encodings are: Value Meaning 00b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint when a match is encountered in the S-mask. 01b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint when a match is encountered in the C-mask.
	0	**Reserved.** Bit reserved for future use and should be set to zero.

60.4.3.4.4 siTD Buffer Pointer List (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4K (page) aligned buffer pointers. The least significant 12 bits of each DWord are used as additional transfer state.

Table 60-75. Buffer Page Pointer List (plus)

Bit	Description
31–12	**Buffer Pointer List.** Bits [31–12] of DWords 4 and 5 are 4K paged aligned, physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. The field *P* specifies the *current* active pointer
11–0	Page 0: **Current Offset.** The 12 least significant bits of the Page 0 pointer is the current byte offset for the current page pointer (as selected with the page indicator bit (*P* field)). The host controller is not required to write this field back when the siTD is retired (*Active* bit transitioned from a one to a zero). The least significant bits of Page 1 pointer is split into three sub-fields Page 1:

	Bits	Description	
	11:5	Reserved.	

Table 60-75. Buffer Page Pointer List (plus)

	4:3	**Transaction position (TP).** This field is used with T-count to determine whether to send *all*, *first*, *middle*, or *last* with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are: Value Meaning 00b All. The entire full-speed transaction data payload is in this transaction (i.e. less than or equal to 188 bytes). 01b Begin. This is the first data payload for a full-speed that is greater than 188 bytes.transaction 10B Mid. This is the *middle* payload for a full-speed OUT transaction that is larger than 188 bytes. 11b End. This is the *last* payload for a full-speed OUT transaction that was larger than 188 bytes.
	2:0	**Transaction count (T-Count).** Software initializes this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

60.4.3.4.5 siTD Back Link Pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD. This pointer cannot reference any other schedule data structure.

Table 60-76. siTD Back Link Pointer

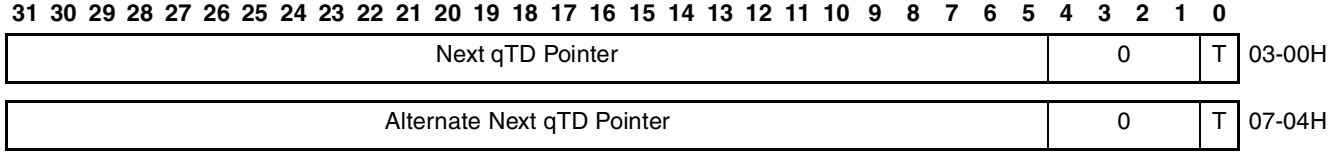
Bit	Description
31–5	**siTD Back Pointer.** This field is a physical memory pointer to a siTD.
4–1	**Reserved.** This field is reserved for future use. It should be set to zero.
0	**Terminate (T).** 1 = siTD Back Pointer field is not valid. 0 = siTD Back Pointer field is valid.

60.4.3.5 Queue Element Transfer Descriptor (qTD)

This data structure is only used with a queue head. This data structure is used for one or more USB transactions. This data structure is used to transfer up to 20480 (5*4096) bytes. The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers. This structure is 32 bytes (or one 32-byte cache line). This data structure must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a queue head.



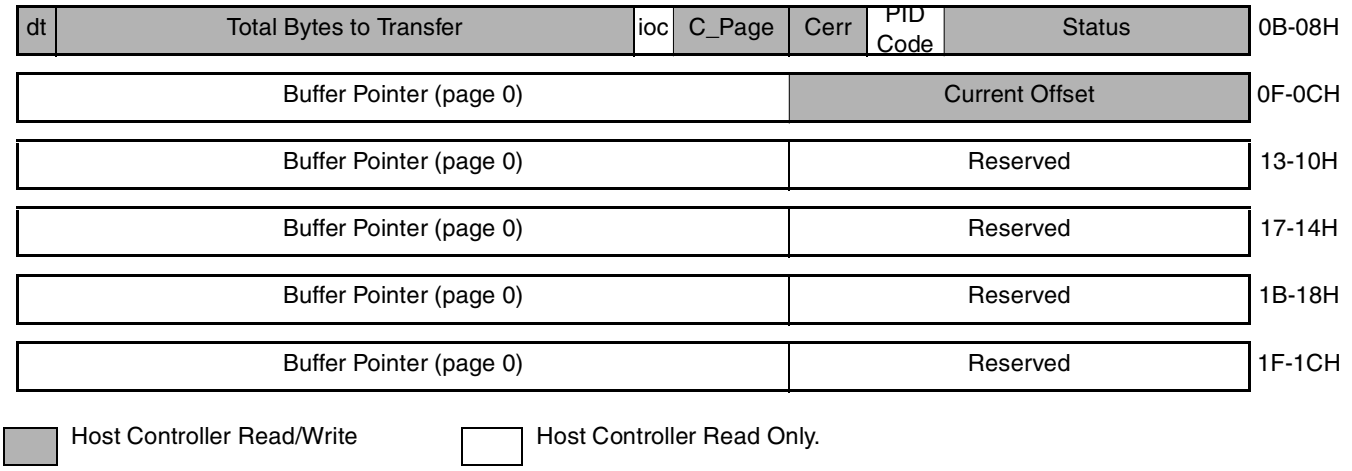


Figure 60-62. Queue Element Transfer Descriptor Block Diagram

Queue Element Transfer Descriptors must be aligned on 32-byte boundaries.

60.4.3.5.1 Next qTD Pointer

The first DWord of an element transfer descriptor is a pointer to another transfer element descriptor.

Table 60-77. qTD Next Element Transfer Pointer (DWord 0)

Bit	Description
31–5	**Next Transfer Element Pointer.** This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals[31:5], respectively.
4–1	**Reserved.** These bits are reserved and their value has no effect on operation.
0	**Terminate (T).** 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

60.4.3.5.2 Alternate Next qTD Pointer

The second DWord of a queue element transfer descriptor is used to support hardware-only advance of the data stream to the next client buffer on short packet. To be more explicit the host controller always uses this pointer when the current qTD is retired due to short packet.

Table 60-78. TD Alternate Next Element Transfer Pointer (DWord 1)

Bit	Description
31–5	**Alternate Next Transfer Element Pointer.** This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31–5], respectively.
4–1	**Reserved.** These bits are reserved and their value has no effect on operation.
0	**Terminate (T).** 1= pointer is invalid. 0 = Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

60.4.3.5.3 qTD Token

The third DWord of a queue element transfer descriptor contains most of the information the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the queue head).

NOTE

The field descriptions forward reference fields defined in the queue head. Where necessary, these forward references are preceded with a QH notation.

Table 60-79. qTD Token (DWord 2)

Bit	Description
31	**Data Toggle.** This is the data toggle sequence bit. The use of this bit depends on the setting of the *Data Toggle Control* bit in the queue head.
30–16	**Total Bytes to Transfer.** This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction, only on the successful completion of the transaction. The maximum value software may store in this field is 5 * 4K (5000H). This is the maximum number of bytes 5 page pointers can access. If the value of this field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the host controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for OUT transfers that *Total Bytes To Transfer* be an even multiple of QHD.Maximum Packet Length. If software builds such a transfer descriptor for an OUT transfer, the last transaction will always be less than QHD.Maximum Packet Length. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. **Therefore, the maximum recommended transfer is 16K(4000H).**
15	**Interrupt On Complete (IOC).** If this bit is set to a one, it specifies that when this qTD is completed, the Host Controller should issue an interrupt at the next interrupt threshold.
14–12	**Current Page (C_Page).** This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0H to 4H. The host controller is not required to write this field back when the qTD is retired.
11–10	**Error Counter (CERR).** This field is a 2-bit down counter that keeps track of the number of consecutive Errors detected while executing this qTD. If this field is programmed with a non-zero value during set-up, the Host Controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from one to zero, the Host Controller marks the qTD inactive, sets the *Halted* bit to a one, and error status bit for the error that caused *CERR* to decrement to zero. An interrupt will be generated if the *USB Error Interrupt Enable* bit in the USBINTR register is set to a one. If HCD programs this field to zero during set-up, the Host Controller will not count errors for this qTD and there will be no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the queue head overlay area, not the qTD. **Error Decrement Counter** Transaction Error Yes Data Buffer Error No[3] Stalled No[1] Babble Detected No[1] No Error No[2]
	Error: Decrement Counter Error Decrement Counter
	1: Detection of Babble or Stall automatically halts the queue head. Thus, count is not decremented

Table 60-79. qTD Token (DWord 2)

	2	If the EPS field indicates a HS device or the queue head is in the Asynchronous Schedule (and *PIDCode* indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller should reset *CERR* to extend the total number of errors for this transaction. For example, *CERR* should be reset with maximum value (3) on each successful completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b. See section Split Transaction Interrupt for CERR adjustment rules when the EPS field indicates a FS or LS device and the queue head is in the Periodic Schedule. See section Asynchronous—Do Complete Split for CERR adjustment rules when the EPS field indicates a FS or LS device, the queue head is in the Asynchronous schedule and the *PIDCode* indicates a SETUP.
	3	Data buffer errors are host problems. They don't count against the device's retries.
	Note: Software must not program CERR to a value of zero when the EPS field is programmed with a value indicating a Full- or Low-speed device. This combination could result in undefined behavior.	
9:8	**PID Code.** This field is an encoding of the token, which should be used for transactions associated with this transfer descriptor. Encodings are:	
	00b	OUT Token generates token (E1H)
	01b	IN Token generates token (69H)
	10b	SETUP Token generates token (2DH) (undefined if endpoint is an Interrupt the queue head is non-zero.) transfer type, for example, µ*Frame S-mask* field in
	11b	Reserved
7:0	**Status.** This field is used by the Host Controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are:	
	Bit	Status Field Description
	7	**Active.** Set to one by software to enable the execution of transactions by the Host Controller.
	6	**Halted.** Set to a one by the Host Controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time that a transaction results in the Halted bit being set to a one, the Active bit is also set to zero.
	5	**Data Buffer Error.** Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, the Host Controller will force a timeout condition on the USB, invalidating the transaction at the source. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	4	**Babble Detected.** Set to a one by the Host Controller during status update when" babble" is detected during the transaction. In addition to setting this bit, the Host Controller also sets the *Halted* bit to a one. Since "babble" is considered a fatal error for the transfer, setting the Halted bit to a one insures that no more transactions occur because of this descriptor.
	3	**Transaction Error (XactErr).** Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.

Table 60-79. qTD Token (DWord 2)

	2	**Missed Micro-Frame.** This bit is ignored unless the *QH.EPS* field indicates a full- or low-speed endpoint and the queue head is in the periodic list. This bit is set when the host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
	1	**Split Transaction State (SplitXstate).** This bit is ignored by the host controller unless the *QH.EPS* field indicates a full- or low-speed endpoint. When a Full- or Low-speed device, the host controller uses this bit to track the state of the split- transaction. The functional requirements of the host controller for managing this state bit and the split transaction protocol depends on whether the endpoint is in the periodic or asynchronous schedule. The bit encodings are: ValueMeaning 0b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint. 1b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint.
	0	**Ping State (P)/ERR.** If the *QH.EPS* field indicates a High-speed device and the *PID_Code* indicates an OUT endpoint, then this is the state bit for the Ping protocol. The bit encodings are: ValueMeaning 0b Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b Do Ping. This value directs the host controller to issue a PING PID to the endpoint. If the *QH.EPS* field does not indicate a High-speed device, then this field is used as an error indicator bit. It is set to a one by the host controller whenever a periodic split-transaction receives an ERR handshake.

60.4.3.5.4 qTD Buffer Page Pointer List

The last five DWords of a queue element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

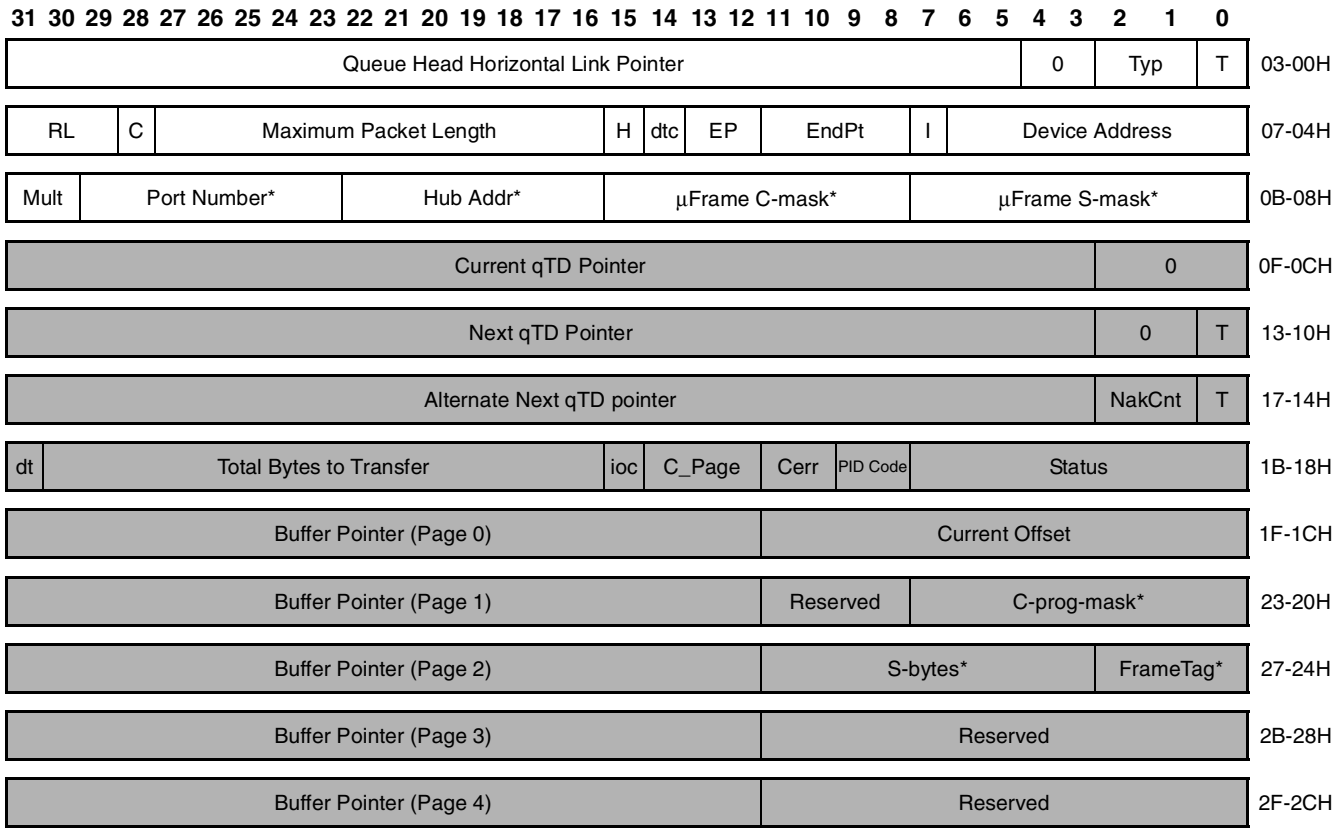
System software initializes *Current Offset* field to the starting offset into the current page, where current page is selected via the value in the *C_Page* field.

Table 60-80. qTD Buffer Pointer(s) (DWords 3–7)

Bit	Description
31–12	**Buffer Pointer List.** Each element in the list is a 4K page aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4K page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to select an array element). If a transaction spans a 4K buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page and advance to the next buffer pointer in the list, and conclude the transaction via the new buffer pointer.
11–0	**Current Offset (Reserved).** This field is reserved in all pointers except the first one (for example, Page 0). The host controller should ignore all reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. Software should ensure the Reserved fields are initialized to zeros.

60.4.3.6 Queue Head

Figure 60-63 shows the queue head structure layout.



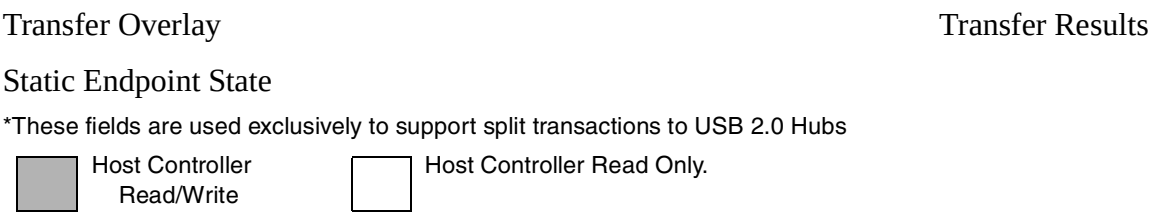


Figure 60-63. Queue Head Structure Layout

Queue Head Horizontal Link Pointer

The first DWord of a Queue Head contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a queue head or one of the isochronous transfer descriptors. It must not reference a queue element transfer descriptor.

Table 60-81. Queue Head DWord 0

Bit	Description
31–5	**Queue Head Horizontal Link Pointer (QHLP).** This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4–3	**Reserved.** These bits must be written as zeros.
2–1	**QH/(s)iTD Select (Typ).** This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: ValueMeaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)
0	**Terminate (T).** 1=Last QH (pointer is invalid). 0=Pointer is valid. If the queue head is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the queue head is in the Asynchronous schedule. Software must ensure that queue heads reachable by the host controller always have valid horizontal link pointers.

60.4.3.6.1 Endpoint Capabilities/Characteristics

The second and third DWords of a Queue Head specifies static information about the endpoint. This information does not change over the lifetime of the endpoint. There are three types of information in this region:

- Endpoint Characteristics. These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint Capabilities. These are adjustable parameters of the endpoint. They effect how the endpoint data stream is managed by the host controller.
- Split Transaction Characteristics. This data structure is used to manage full- and low-speed data streams for bulk, control, and interrupt via split transactions to USB2.0 Hub Transaction Translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

Table 60-82. Endpoint Characteristics: Queue Head DWord 1

Bit	Description
31–28	**Nak Count Reload (RL).** This field contains a value, which is used by the host controller to reload the Nak Counter field.
27	**Control Endpoint Flag (C).** If the *QH.EPS* field indicates the endpoint is not a high-speed device, and the endpoint is a control endpoint, then software must set this bit to a one. Otherwise, it should always set this bit to a zero.
26–16	**Maximum Packet Length.** This directly corresponds to the maximum packet size of the associated endpoint (*wMaxPacketSize*). The maximum value this field may contain is 0x400 (1024).
15	**Head of Reclamation List Flag (H).** This bit is set by System Software to mark a queue head as being the head of the reclamation list.
14	**Data Toggle Control (DTC).** This bit specifies where the host controller should get the initial data toggle on an overlay transition. 0b Ignore DT bit from incoming qTD. Host controller preserves DT bit in the queue head. 1b Initial data toggle comes from incoming qTD DT bit. Host controller replaces DT bit in the queue head from the DT bit in the qTD.
13–12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are: Value – Meaning 00b – Full-Speed (12Mbs) 01b – Low-Speed (1.5Mbs) 10b – High-Speed (480 Mb/s) 11b – Reserved This field must not be modified by the host controller.
11–8	**Endpoint Number (Endpt).** This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	**Inactivate on Next Transaction (I).** This bit is used by system software to request that the host controller set the Active bit to zero. See Section Rebalancing the Periodic Schedule for full operational details. This field is only valid when the queue head is in the Periodic Schedule and the *EPS* field indicates a Full or Low-speed endpoint. Setting this bit to a one when the queue head is in the Asynchronous Schedule or the *EPS* field indicates a high-speed device yields undefined results.
6–0	**Device Address.** This field selects the specific device serving as the data source or sink.

Table 60-83. Endpoint Capabilities: Queue Head DWord 2

Bit	Description
31–30	**High-Bandwidth Pipe Multiplier (Mult).** This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that software properly initializes this field (regardless of location of queue head in the schedules or other run time parameters). The valid values are: ValueMeaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame 10b Two transactions to be issued for this endpoint per micro-frame 11b Three transactions to be issued for this endpoint per micro-frame
29–23	**Port Number.** This field is ignored by the host controller unless the *EPS* field indicates a full- or low-speed device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address *Hub Addr* below), below which the full- or low-speed device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22–16	**Hub Addr.** This field is ignored by the host controller unless the *EPS* field indicates a full-or low-speed device. The value is the USB device address of the USB 2.0 Hub below which the full- or low-speed device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15–8	**Split Completion Mask (μFrame C-Mask).** This field is ignored by the host controller unless the *EPS* field indicates this device is a low- or full-speed device and this queue head is in the periodic list. This field (along with the *Active* and *SplitX-state* fields) is used to determine during which micro-frames the host controller should execute a complete-split transaction. When the criteria for using this field are met, a zero value in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of the FRINDEX register. If the FRINDEX register bits decode to a position where the μ*Frame C- Mask* field is a one, then this queue head is a candidate for transaction execution. There may be more than one bit in this mask set.
7–0	**Interrupt Schedule Mask (μFrame S-mask).** This field is used for all endpoint speeds. Software should set this field to a zero when the queue head is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of the FRINDEX register as an index into a bit position in this bit vector. If the μ*Frame S-mask* field has a one at the indexed bit position then this queue head is a candidate for transaction execution. If the *EPS* field indicates the endpoint is a high-speed endpoint, then the transaction executed is determined by the *PID_Code* field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the *EPS* field indicates this is either a full- or low-speed device. A zero value in this field, in combination with existing in the periodic frame list has undefined results.

60.4.3.6.2 Transfer Overlay

The nine DWords in this area represent a *transaction working space* for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the *Queue Head Horizontal Link Pointer* to the next queue head. The host controller will never follow the *Next Transfer Queue Element or Alternate Queue Element* pointers unless it is actively attempting to advance the queue. For the duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a Queue Head contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

Table 60-84. Current qTD Link Pointer

Bit	Description
31–5	**Current Element Transaction Descriptor Link Pointer.** This field contains the address Of the current transaction being processed in this queue and corresponds to memory address signals [31–5], respectively.
4–0	**Reserved (R).** These bits are ignored by the host controller when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4–11 of a queue head are the transaction overlay area. This area has the same base structure as a Queue Element Transfer Descriptor. The queue head utilizes the reserved fields of the page pointers to implement tracking the state of split transactions.

This area is characterized as an *overlay* because when the queue is advanced to the next queue element, the source queue element is *merged* onto this area. This area serves an execution cache for the transfer.

Table 60-85. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9)

DWord	Bit	Description
5	4:1	**Nak Counter (NakCnt)µRW**. This field is a counter the host controller decrements whenever a transaction for the endpoint associated with this queue head results in a Nak or Nyet response. This counter is reloaded from *RL* before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from RL during an overlay.
6	31	**Data Toggle.** The *Data Toggle Control* controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	**Interrupt On Complete (IOC).** The IOC control bit is always inherited from the source qTD when the overlay operation is performed.
6	11:10	**Error Counter (C_ERR).** This two-bit field is copied from the qTD during the overlay and written back during queue advancement.
6	0	**Ping State (P)/ERR.** If the *EPS* field indicates a high-speed endpoint, then this field should be preserved during the overlay operation.
8	7:0	**Split-transaction Complete-split Progress (C-prog-mask).** This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4:0	**Split-transaction Frame Tag (Frame Tag).** This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	11:5	**S-bytes.** Software must ensure that the *S-bytes* field in a *qTD* is zero before activating the *qTD*. This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

60.4.3.7 Periodic Frame Span Traversal Node (FSTN)

This data structure is to be used only for managing Full- and Low-speed transactions that span a Host-frame boundary. See section Host Controller Operational Model for FSTNs for full operational details. Software must not use an FSTN in the Asynchronous Schedule. An FSTN in the Asynchronous schedule results in undefined behavior. Software must not use the FSTN feature with a host controller whose HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use will yield undefined results.

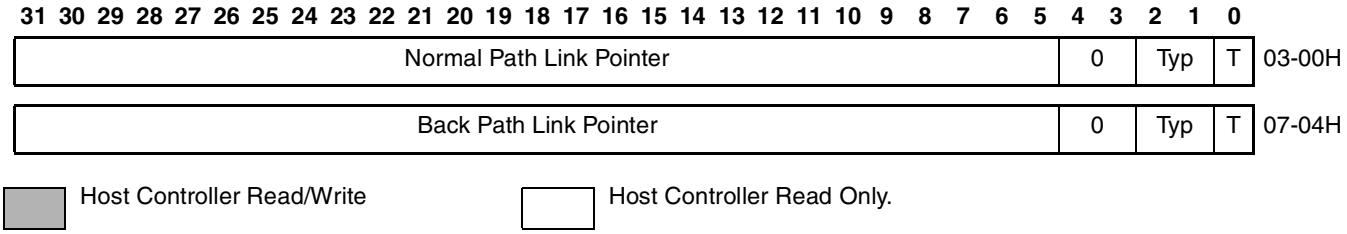


Figure 60-64. Frame Span Traversal Node Structure Layout

60.4.3.7.1 FSTN Normal Path Pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

Bit	Description
31:5	**Normal Path Link Pointer (NPLP).** This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4:3	Reserved. These bits must be written as 0s.
2:1	**QH/(s)iTD/FSTN Select (Typ).** This field indicates to the Host Controller whether the item referenced is a iTD/siTD, a QH or an FSTN. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are: **ValueMeaning** 00biTD (isochronous transfer descriptor) 01bQH (queue head) 10bsiTD (split transaction isochronous transfer descriptor) 11bFSTN (Frame Span Traversal Node)
0	**Terminate (T).** 1=Link Pointer field is not valid. 0=Link Pointer is valid.

60.4.3.7.2 FSTN Back Path Link Pointer

The second DWord of an FTSN node contains a link pointer to a queue head. If the *T-bit* in this pointer is a zero, then this FSTN is a *Save-Place* indicator. Its *Typ* field must be set by software to indicate the target

data structure is a queue head. If the *T-bit* in this pointer is set to a one, then this FSTN is the *Restore* indicator. When the *T-bit* is a one, the host controller ignores the *Typ* field.

Table 60-86. FSTN Back Path Link Pointer Field Descriptions

Bit	Description
31–5	**Back Path Link Pointer (BPLP).** This field contains the address of a Queue Head. This field corresponds to memory address signals [31:5], respectively.
4–3	**Reserved.** These bits must be written as 0s.
2–1	**Typ.** Software must ensure this field is set to indicate the target data structure is a Queue Head. Any other value in this field yields undefined results.
0	**Terminate (T).** 1 Link Pointer field is not valid (i.e. the host controller must not use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Restore indicator. 0 Link Pointer is valid (i.e. the host controller may use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Save-Place indicator.

60.4.4 Host Operational Model

The general operational model is for the enhanced interface host controller hardware and enhanced interface host controller driver (generally referred to as system software). Each significant operational feature of the EHCI host controller is discussed in a separate section. Each section presents the operational model requirements for the host controller hardware. Where appropriate, recommended system software operational models for features are also presented.

60.4.4.1 Host Controller Initialization

When the system boots, the host controller is enumerated, assigned a base address for the register space and BIOS sets the FLADJ register to a system-specific value. After initial power-on or *HCReset* (hardware or via *HCReset* bit in the USBCMD register), all of the operational registers will be at their default values, as illustrated in Table 60-87. After a hardware reset, only the operational registers not contained in the Auxiliary power well will be at their default values.

Table 60-87. Default Values of Operational Register Space

Operational Register	Default Value (after Reset)
USBCMD	00080000h (00080B00h if *Asynchronous Schedule Park Capability* is a one)
USBSTS	00001000h
USBINTR	00000000h
FRINDEX	00000000h
CTRLDSSEGMENT	00000000h
PERIODICLISTBASE	Undefined
ASYNCLISTADDR	Undefined

Table 60-87. Default Values of Operational Register Space (continued)

CONFIGFLAG	00000000h
PORTSC	00002000h (w/*PPC* set to one); 00003000h (w/*PPC* set to a zero)

In order to initialize the host controller, software should perform the following steps

- Program the CTRLDSSEGMENT register with 4-Gigabyte segment where all of the interface data structures are allocated.
- Write the appropriate value to the USBINTR register to enable the appropriate interrupts.
- Write the base address of the Periodic Frame List to the PERIODICLIST BASE register. If there are no work items in the periodic schedule, all elements of the Periodic Frame List should have their *T-Bits* set to a one.
- Write the USBCMD register to set the desired interrupt threshold, frame list size (if applicable) and turn the host controller *ON* via setting the *Run/Stop* bit.
- Write a 1 to CONFIGFLAG register to route all ports to the EHCI controller.

At this point, the host controller is up and running and the port registers will begin reporting device connects, etc. System software can enumerate a port through the reset process (where the port is in the enabled state). At this point, the port is active with SOFs occurring down the enabled port enabled High-speed ports, but the schedules have not yet been enabled. The EHCI Host controller will not transmit SOFs to enabled Full- or Low-speed ports. In order to communicate with devices via the asynchronous schedule, system software must write the ASYNDLISTADDR register with the address of a control or bulk queue head. Software must then enable the asynchronous schedule by writing a one to the *Asynchronous Schedule Enable* bit in the USBCMD register. In order to communicate with devices via the periodic schedule, system software must enable the periodic schedule by writing a one to the *Periodic Schedule Enable* bit in the USBCMD register. Note that the schedules can be turned on before the first port is reset (and enabled).

Any time the USBCMD register is written, system software must ensure the appropriate bits are preserved, depending on the intended operation.

60.4.4.2 Port Routing and Control

A USB 2.0 Host controller is comprised of one high-speed host controller, which implements the EHCI programming interface and 0 to N USB 1.1 companion host controllers. Companion host controllers (cHCs) may be implementations of either Universal or Open host controller specifications. This configuration is used to deliver the required full USB 2.0-defined port capability; for example, Low-, Full-, and High-speed capability for every port. Figure 60-65 illustrates a simple block diagram of the port

routing logic and its relationship to the high-speed and companion host controllers within a USB 2.0 host controller.

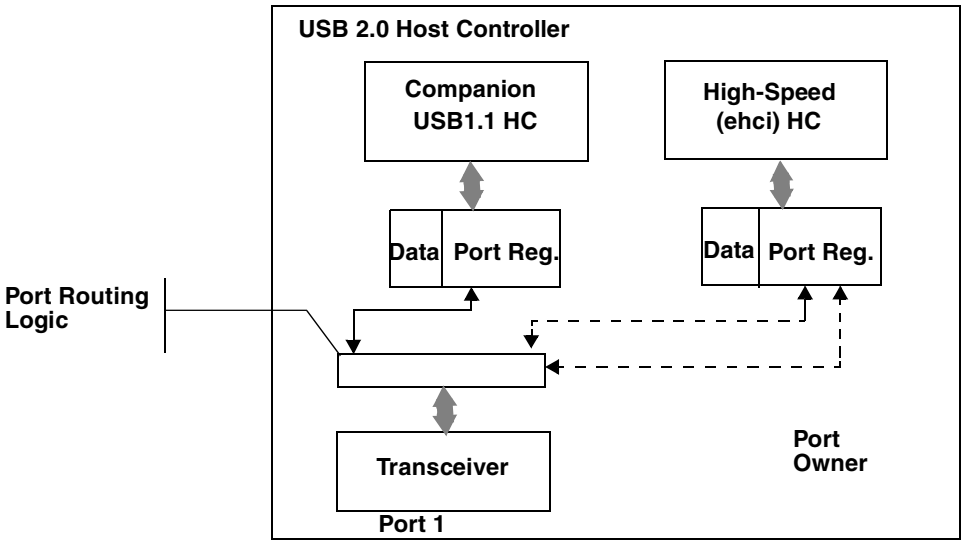


Figure 60-65. Example USB 2.0 Host Controller Port Routing Block Diagram

There exists one transceiver per physical port and each host controller module has its own port status and control registers. The EHCI controller has port status and control registers for every port. Each companion host controller has only the port control and status registers it is required to operate. Each transceiver can be controlled by either the EHCI host controller or one companion host controller. Routing logic lies between the transceiver and the port status and control registers.

NOTE

The routing logic should not be implemented in the 480 MHz clock domain of the transceiver.

The port routing logic is controlled from signals originating in the EHCI host controller. The EHCI host controller has a *global* routing policy control field and per-port *ownership* control fields. The *Configured Flag* (*CF*) bit (defined in section BURSTSIZE) is the global routing policy control. At power-on or reset, the default routing policy is to the companion controllers (if they exist). If the system does not include a driver for the EHCI host controller and the host controller includes Companion Controllers, then the ports will still work in Full- and Low-speed mode (assuming the system includes a driver for the companion controllers). In general, when the EHCI owns the ports, the companion host controllers' port registers do not see a connect indication from the transceiver. Similarly, when a companion host controller owns a port, the EHCI controller's port registers do not see a connect indication from the transceiver. The details on the rules for the port routing logic are described in the following sections. The USB 2.0 host controller must be implemented as a multi-function PCI device if the implementation includes companion controllers. The companion host controllers' function numbers must be less than the EHCI host controller function number. The EHCI host controller must be a larger function number with respect to the companion host controllers associated with this EHCI host controller. If a PCI device implementation contains only an EHCI controller (i.e. no companion controllers or other PCI functions), then the EHCI host controller must be function zero, in accordance with the PCI Specification. The *N_CC* field in the Structural Parameter

register (*HCSPARAMS*) indicates whether the controller implementation includes companion host controllers. When *N_CC* has a non-zero value there exists companion host controllers. If N_CC has a value of zero, then the host controller implementation does not include companion host controllers. If the host controller root ports are exposed to attachment of full- or low-speed devices, the ports will always fail the high-speed chirp during reset and the ports will not be enabled. System software can notify the user of the illegal condition. This type of implementation requires a USB 2.0 hub be connected to a root port to provide full and low-speed device connectivity.

System software uses information in the host controller capability registers to determine how the ports are routed to the companion host controllers. See Section 60.4.2.4.3, HCSPARAMS – EHCI Compliant with extensions."

NOTE

If an implementation includes more than one set of companion and EHCI host controllers, they are organized as groups of companion host controllers with intermixed EHCI controllers.

60.4.4.2.1 Port Routing Control via EHCI *Configured* (*CF*) Bit

Each port in the USB 2.0 host controller can be routed either to a single companion host controller or to the EHCI host controller. The port routing logic is controlled by two mechanisms in the EHCI HC: a host controller global flag and per-port control. The *Configured Flag* (*CF*) bit (defined in Section BURSTSIZE), is used to globally set the policy of the routing logic. Each port register has a *Port Owner* control bit which allows the EHCI Driver to explicitly control the routing of individual ports. Whenever the *CF bit* transitions from a zero to a one (this transition is only available under program control) the port routing unconditionally routes all of the port registers to the EHCI HC (all *Port Owner* bits go to zero). While the *CF-bit* is a one, the EHCI Driver can control individual ports' routing via the *Port Owner* control bit. Likewise, whenever the *CF bit* transitions from a one to a zero (as a result of Aux power application, *HCRESET*, or software writing a zero to *CF-bit*), the port routing unconditionally routes all of the port registers to the appropriate companion HC. The default value for the EHCI HC's *CF bit* (after Aux power

application or *HCRESET*) is zero. Table 60-88 summarizes the default routing for all the ports, based on the value of the EHCI HC's *CF bit*.

The *view* of the port depends on the current owner. A Universal or Open companion host controller will see port register bits consistent with the appropriate specification. Port bit definitions that are required for EHCI host controllers are not visible to companion host controllers.

Table 60-88. Default Port Routing Depending on EHCI HC CF Bit

HS CF Bit	Default Port Ownership	Explanation
0B	Companion HCs	The companion host controllers own the ports and only Full- and Low-speed devices are supported in the system. The exact port assignments are implementation dependent. The ports behave only as Full- and Low-speed ports in this configuration
1B	EHCI HC	The EHCI host controller has default ownership over all of the ports. The routing logic inhibits device connect events from reaching the companion HCs' port status and control registers when the port owner is the EHCI HC.The EHCI HC has access to the additional port status and control bits defined in this specification (see Section PORTSCx—Port Status Control[1:8]). The EHCI HC can temporarily release control of the port to a companion HC by setting the *PortOwner* bit in the PORTSC register to a one.

60.4.4.2.2 Port Routing Control via *PortOwner* and Disconnect Event

Manipulating the port routing via the *CF-bit* is an extreme process and not intended to be used during normal operation. The normal mode of port ownership transferal is on the granularity of individual ports using the *Port Owner* bit in the EHCI HC's PORTSC register (for hand-offs from EHCI to companion host controllers). Individual port ownership is returned to the EHCI controller when the port registers a device disconnect. When the disconnect is detected, the port routing logic immediately returns the port ownership to the EHCI controller. The companion host controller port register detects the device disconnect and operates normally.

Under normal operating conditions (assuming all HC drivers loaded and operational and the EHCI *CF-bit* is set to a one), the typical port enumeration sequence proceeds as illustrated below:

- Initial condition is that EHCI is port owner. A device is connected causing the port to detect a connect, set the port connect change bit and issue a port-change interrupt (if enabled).
- EHCI Driver identifies the port with the new connect change bit asserted and sends a change report to the hub driver. Hub driver issues a GetPortStatus() request and identifies the connect change. It then issues a request to clear the connect change, followed by a request to reset and enable the port.
- When the EHCI Driver receives the request to reset and enable the port, it first checks the value reported by the *LineStatus* bits in the PORTSC register. If they indicate the attached device is a full-speed device (for example, D+ is asserted), then the EHCI Driver sets the *PortReset* control bit to a one (and sets the *PortEnable* bit to a zero) which begins the reset-process. Software times the duration of the reset, then terminates reset signaling by writing a zero to the port reset bit. The reset process is actually complete when software reads a zero in the *PortReset* bit. The EHCI Driver checks the PortOwner bit in the PORTSC register. If set to a one, the connected device is a high-speed device and EHCI Driver (root hub emulator) issues a change report to the hub driver and the hub driver continues to enumerate the attached device.

- At the time the EHCI Driver receives the port reset and enable request the *LineStatus* bits might indicate a low-speed device. Additionally, when the port reset process is complete, the *PortEnable* field may indicate that a full-speed device is attached. In either case the EHCI driver sets the PortOwner bit in the PORTSC register to a one to release port ownership to a companion host controller.
- When the EHCI Driver sets *PortOwner* bit to a one, the port routing logic makes the connection state of the transceiver available to the companion host controller port register and removes the connection state from the EHCI HC port. The EHCI PORTSC register observes and reports a disconnect event via the disconnect change bit. The EHCI Driver detects the connection status change (either by polling or by port change interrupt) and then sends a change report to the hub driver. When the hub driver requests that port-state, the EHCI Driver responds with a reset complete change set to a one, a connect change set to a one and a connect status set to a zero. This information is derived directly from the EHCI port register. This will allow the hub driver to assume the device was disconnected during reset. It will acknowledge the change bits and wait for the next change event. While the EHCI controller does not own the port, it simply remains in a state where the port reports no device connected. The device-connect evaluation circuitry of the companion HC activates and detects the device, the companion Driver detects the connection and enumerates the port.

When a port is routed to a companion HC, it remains under the control of the companion HC until the device is disconnected from the root port (ignoring for now the scenario where EHCI's *CF-bit* transitions from a 1b to a 0b). When a disconnect occurs, the disconnect event is detected by both the companion HC port control and the EHCI port ownership control. On the event, the port ownership is returned immediately to the EHCI controller. The companion HC stack detects the disconnect and acknowledges as it would in an ordinary standalone implementation. Subsequent connects will be detected by the EHCI port register and the process will repeat.

60.4.4.2.3 Example Port Routing State Machine

Figure 60-66 illustrates an example of how the port ownership should be managed. The following sections describe the entry conditions to each state.

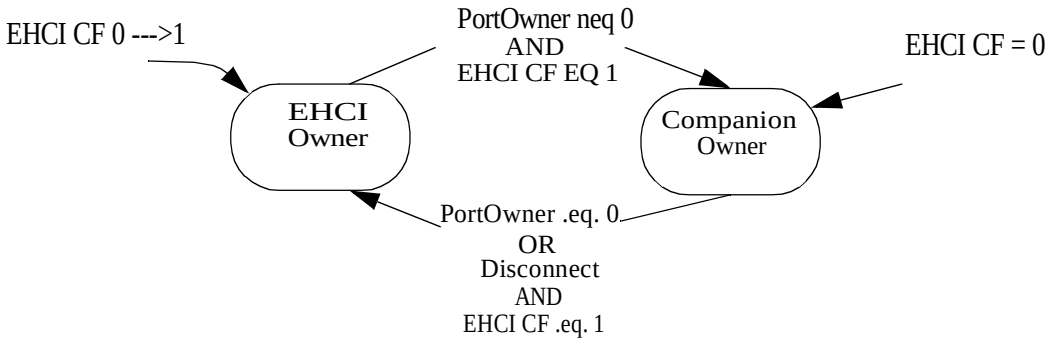


Figure 60-66. Port Owner Handoff State Machine

EHCI HC Owner

Entry to this state occurs whenever one of the following events occur:

- When the EHCI HC's *Configure Flag* (*CF*) bit in the CONFIGFLAG register transitions from a zero to a one. This signals the fact that the system has a host controller driver for the EHCI HC and that all ports in the USB 2.0 host controller must default route to the EHCI controller.
- When the port is owned by a companion HC and the device is disconnected from the port. The EHCI port routing control logic is notified of the disconnect, and returns port routing to the EHCI controller. The connection state of the companion HC goes immediately to the disconnected state (with appropriate side effect to connect change, enable and enable change). The companion HC driver will acknowledge the disconnect by setting the connect status change bit to a zero. This allows the companion HC's driver to interact with the port completely through the disconnect process.
- When system software writes a zero to the *PortOwner* bit in the PORTSC register. This allows software to take ownership of a port from a companion host controller. When this occurs, the routing logic to the companion HC effectively signals a disconnect to the companion HC's port status and control register.

Companion HC Owner

Entry to this state occurs whenever one of the following events occur:

- When the *Port Owner* field transitions from a zero to a one.
- When the HS-mode HC's *Configure Flag* (*CF*) is equal to zero.

On entry to this state, the routing logic allows the companion HC port register to detect a device connect. Normal port enumeration proceeds.

60.4.4.2.4 Port Power

The *Port Power Control* (*PPC*) bit in the HCSPARAMS register indicates whether the USB 2.0 host controller has port power control (See section HCSPARAMS – EHCI Compliant with extensions). When this bit is a zero, then the host controller does not support software control of port power switches. When in this configuration, the port power is always available and the companion host controllers must implement functionality consistent with port power always on. When the *PPC* bit is a one, then the host controller implementation includes port power switches. Each available switch has an output enable, which is referred to in this discussion as *PortPowerOutputEnable* (*PPE*). *PPE* is controlled based on the

state of the combination bits *PPC* bit, EHCI *Configured* (*CF*)-bit and individual *Port Power* (*PP*) bits. Table 60-89 illustrates the summary behavioral model.

Table 60-89. Port Power Enable Control Rules

CF	CHC[2] (PP)	EHC[3] (PP)	Owner	PPE[1]	Description
0	0	X	CHC	0	When the EHCI controller has not been configured, the port is owned by the companion host controller. When the companion HC's port power select is off, then the port power is off.
0	1	X	CHC	1	Similar to previous entry. When the companion HC's port power select is on, then the port power is on.
1	0	0	CHC	0	Port owner has port power turned off, the power to port is off.
1	0	0	EHC	0	Port owner has port power turned off, the power to port is off.
1	0	1	EHC	1	Port owner has port power on, so power to port is on.
1	0	1	CHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	EHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	EHC	1	Port owner has port power on, so power to port is on.

[1]PPE (Port Power Enable). This bit actually turns on the port power switch (if one exists).
[2]CHC (Companion Host Controller).
[3]EHC (EHCI Host Controller).

60.4.4.2.5 Port Reporting Over-Current

Host controllers are by definition power providers on USB. Whether the ports are considered high- or low-powered is a platform implementation issue. Each EHCI PORTSC register has an over-current status and over-current change bit. The functionality of these bits is specified in the USB Specification Revision 2.0.

The over current detection and limiting logic usually resides outside the host controller logic. This logic may be associated with one or more ports. When this logic detects an over-current condition it is made available to both the companion and EHCI ports. The effect of an over-current status on a companion host controller port is beyond the scope of this document. The over-current condition effects the following bits in the PORTSC register on the EHCI port:

- *Over-current Active* bits are set to a one. When the over-current condition goes away, the *Over-current Active* bit will transition from a one to a zero.
- *Over-current Change* bits are set to a one. On every transition of the *Over-current Active* bit the host controller will set the *Over-current Change* bit to a one. Software sets the *Over-current Change* bit to a zero by writing a one to this bit.
- *Port Enabled/Disabled* bit is set to a zero. When this change bit gets set to a one, then the *Port Change Detect* bit in the USBSTS register is set to a one.

- *Port Power (PP)* bits may optionally be set to a zero. There is no requirement in USB that a power provider shut off power in an over current condition. It is sufficient to *limit* the current and leave power applied. When the *Over-current Change* bit transitions from a zero to a one, the host controller also sets the *Port Change Detect* bit in the USBSTS register to a one. In addition, if the *Port Change Interrupt Enable* bit in the USBINTR register is a one, then the host controller will issue an interrupt to the system. Refer to Table 60-90 for summary behavior for over-current detection when the host controller is halted (suspended from a device component point of view).

60.4.4.3 Suspend/Resume

The EHCI host controller provides an equivalent suspend and resume model as that defined for individual ports in a USB 2.0 Hub. Control mechanisms are provided to allow system software to suspend and resume individual ports. The mechanisms allow the individual ports to be resumed completely via software initiation. Other control mechanisms are provided to parameterize the host controller's response (or sensitivity) to external resume events. In this discussion, host-initiated, or software initiated resumes are called Resume Events/Actions. Bus-initiated resume events are called wake-up events. The classes of wakeup events are:

- Remote-wakeup enabled device asserts resume signaling. In similar kind to USB 2.0 Hubs, EHCI controllers must always respond to explicit device resume signaling and wake up the system (if necessary).
- Port connect and disconnect and over-current events. Sensitivity to these events can be turned on or off by using the per-port control bits in the PORTSC registers.

Selective suspend is a feature supported by every PORTSC register. It is used to place specific ports into a suspend mode. This feature is used as a functional component for implementing the appropriate power management policy implemented in a particular operating system. When system software intends to suspend the entire bus, it should selectively suspend all enabled ports, then shut off the host controller by setting the *Run/Stop* bit in the USBCMD register to a zero. The EHCI module can then be placed into a lower device state via the PCI power management interface (See Appendix A, Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com).

When a wake event occurs the system will resume operation and system software will eventually set the *Run/Stop* bit to a one and resume the suspended ports. Software must not set the *Run/Stop* bit to a one until it is confirmed that the clock to the host controller is stable. This is usually confirmed in a system implementation in that all of the clocks in the system are stable before the CPU is restarted. So, by definition, if software is running, clocks in the system are stable and the *Run/Stop* bit in the USBCMD register can be set to a one. There are also minimum system software delays defined in the PCI Power Management Specification. Refer to this specification for more information.

60.4.4.3.1 Port Suspend/Resume

System software places individual ports into suspend mode by writing a one into the appropriate PORTSC *Suspend* bit. Software must only set the *Suspend* bit when the port is in the enabled state (*Port Enabled* bit is a one) and the EHCI is the port owner (*Port Owner* bit is a zero).

The host controller may evaluate the *Suspend* bit immediately or wait until a micro-frame or frame boundary occurs. If evaluated immediately, the port is not suspended until the current transaction (if one is executing) completes. Therefore, there may be several micro-frames of activity on the port until the host controller evaluates the *Suspend* bit. The host controller must evaluate the *Suspend* bit at least every frame boundary.

System software can initiate a resume on a selectively suspended port by writing a one to the *Force Port Resume* bit. Software should not attempt to resume a port unless the port reports that it is in the suspended state (see Section PORTSCx). If system software sets *Force Port Resume* bit to a one when the port is not in the suspended state, the resulting behavior is undefined. In order to assure proper USB device operation, software must wait for at least 10 milliseconds after a port indicates that it is suspended (*Suspend* bit is a one) before initiating a port resume via the *Force Port Resume* bit. When *Force Port Resume* bit is a one, the host controller sends resume signaling down the port. System software times the duration of the resume (nominally 20 milliseconds) then sets the *Force Port Resume* bit to a zero. When the host controller receives the write to transition *Force Port Resume* to zero, it completes the resume sequence as defined in the USB specification, and sets both the *Force Port Resume* and *Suspend* bits to zero. Software-initiated port resumes do not affect the *Port Change Detect* bit in the USBSTS register nor do they cause an interrupt if the *Port Change Interrupt Enable* bit in the USBINTR register is a one. An external USB event may also initiate a resume. The wake events are defined above. When a wake event occurs on a suspended port, the resume signaling is detected by the port and the resume is reflected downstream within 100 µsec. The port's *Force Port Resume* bit is set to a one and the *Port Change Detect* bit in the USBSTS register is set to a one. If the *Port Change Interrupt Enable* bit in the USBINTR register is a one the host controller will issue a hardware interrupt.

System software observes the resume event on the port, delays a port resume time (nominally 20 msec), then terminates the resume sequence by writing zero to the *Force Port Resume* bit in the port. The host controller receives the write of zero to *Force Port Resume*, terminates the resume sequence and sets *Force Port Resume* and *Suspend* port bits to zero. Software can determine that the port is enabled (not suspended) by sampling the PORTSC register and observing that the *Suspend* and *Force Port Resume* bits are zero. Software must ensure that the host controller is running (i.e. *HCHalted* bit in the USBSTS register is a zero), before terminating a resume by writing a zero to a port's *Force Port Resume* bit. If *HCHalted* is a one when *Force Port Resume* is set to a zero, then SOFs will not occur down the enabled port and the device will return to suspend mode in a maximum of 10 milliseconds.

Table 60-90 summarizes the wake-up events. Whenever a resume event is detected, the *Port Change Detect* bit in the USBSTS register is set to a one. If the *Port Change Interrupt Enable* bit is a one in the USBINTR register, the host controller will also generate an interrupt on the resume event. Software acknowledges the resume event interrupt by clearing the *Port Change Detect* status bit in the USBSTS register.

Table 60-90. Behavior During Wake-up Events

Port Status and Signaling Type	Signaled Port Response	Device State	
		D0	not D0
Port disabled, resume K-State received	No Effect	N/A	N/A

Table 60-90. Behavior During Wake-up Events (continued)

Port suspended, Resume K-State received	Resume reflected downstream on signaled port. Force Port Resume status bit in PORTSC register is set to a one. Port Change Detect bit in USBSTS register set to a one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is a one. A disconnect is detected.	Depending in the initial port state, the PORTSC Connected Enable status bits are set to zero, and the Connect Change status bit is set to a one. Port Change Detect bit in the USBSTS register is set to a one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is a zero. A disconnect is detected.	Depending on the initial port state, the PORTSC Connect and Enable status bits are set to zero, and the Connect Change status bit is set to a one. Port Change Detect bit in the USBSTS register is set to a one.	[1], [3]	[3]
Port is not connected and the port's WKCNNT_E bit is a one. A connect is detected.	PORTSC Connect Status and Connect Status Change bits are set to a one. Port Change Detect bit in the USBSTS register is set to a one.	[1], [2]	[2]
Port is not connected and the port's WKCNNT_E bit is a zero. A connect is detected.	PORTSC Connect Status and Connect Status Change bits are set to a one. Port Change Detect bit in the USBSTS register is set to a one.	[1], [3]	[3]
Port is connected and the port's WKOC_E bit is a one. An over-current condition occurs.	PORTSC Over-current Active, Over-current Change bits are set to a one. If Port Enable/Disable bit is a one, it is set to a zero. Port Change Detect bit in the USBSTS register is set to a one	[1], [2]	[2]
Port is connected and the port's WKOC_E bit is a zero. An over-current condition occurs.	PORTSC Over-current Active, Over-current Change bits are set to a one. If Port Enable/Disable bit is a one, it is set to a zero. Port Change Detect bit in the USBSTS register is set to a one.	[1], [3]	[3]

[1] Hardware interrupt issued if Port Change Interrupt Enable bit in the USBINTR register is a one.
[2] PME# asserted if enabled (Note: PME Status must always be set to a one).
[3] PME# not asserted.

60.4.4.4 Schedule Traversal Rules

The host controller executes transactions for devices using a simple, shared-memory schedule. The schedule is comprised of a few data structures, organized into two distinct lists. The data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic and hardware / software complexity.

System software maintains two schedules for the host controller: a periodic schedule and an asynchronous schedule. The root of the periodic schedule is the *PERIODICLISTBASE* register (see Section PERIODICLISTBASE; DEVICEADDR). The *PERIODICLISTBASE* register is the physical memory base address of the periodic frame list. The periodic frame list is an array of physical memory pointers. The objects referenced from the frame list must be valid schedule data structures as defined in Host Data Structure. In each micro-frame, if the periodic schedule is enabled (see Section Periodic Scheduling Threshold) then the host controller must execute from the periodic schedule before executing from the asynchronous schedule. It will only execute from the asynchronous schedule after it encounters the end of the periodic schedule. The host controller traverses the periodic schedule by constructing an array offset

reference from the *PERIODICLISTBASE* and the *FRINDEX* registers (see Figure 60-67). It fetches the element and begins traversing the graph of linked schedule data structures.

The end of the periodic schedule is identified by a *next* link pointer of a schedule data structure having its *T-bit* set to a one. When the host controller encounters a *T-Bit* set to a one during a horizontal traversal of the periodic list, it interprets this as an End-Of-Periodic-List mark. This causes the host controller to cease working on the periodic schedule and transitions immediately to traversing the asynchronous schedule. Once this transition is made, the host controller executes from the asynchronous schedule until the end of the micro-frame.

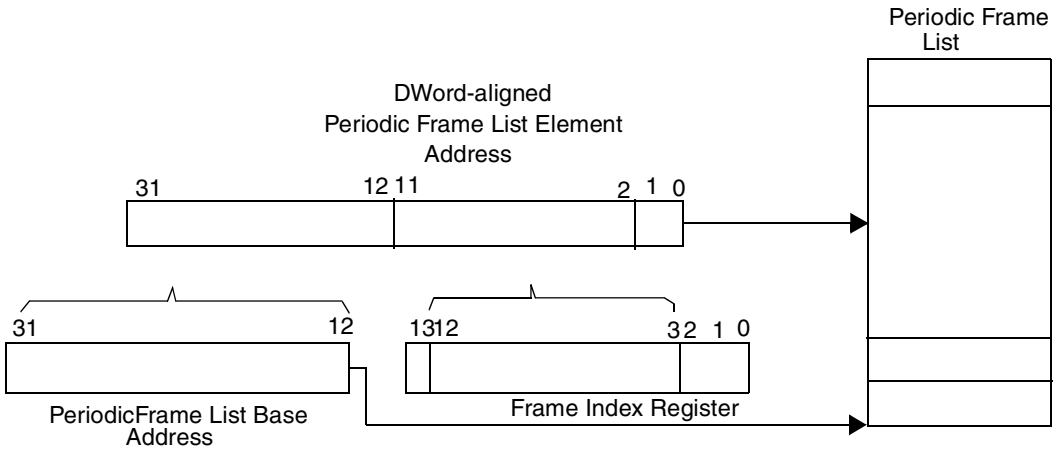


Figure 60-67. Derivation of Pointer into Frame List Array

When the host controller determines that it is time to execute from the asynchronous list, it uses the operational register *ASYNCLISTADDR* to access the asynchronous schedule, see Figure 60-68.

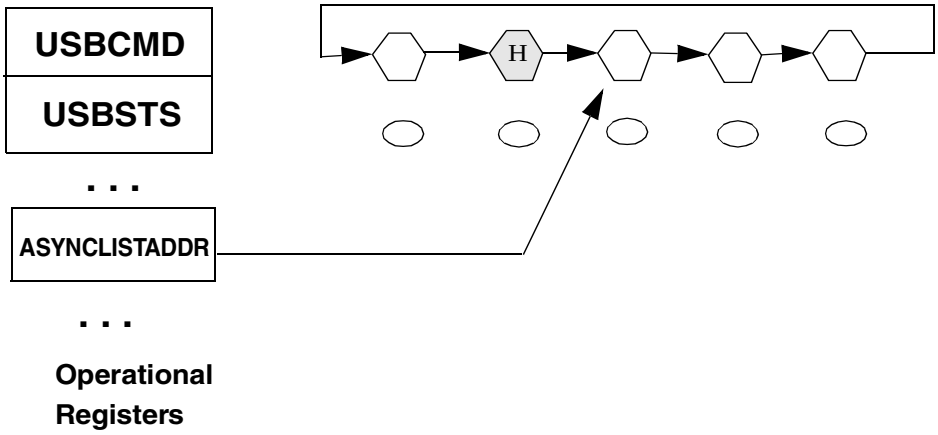


Figure 60-68. General Format of Asynchronous Schedule List

The *ASYNCLISTADDR* register contains a physical memory pointer to the *next* queue head. When the host controller makes a transition to executing the asynchronous schedule, it begins by reading the queue head

referenced by the *ASYNCLISTADDR* register. Software must set queue head *horizontal* pointer *T-bits* to a zero for queue heads in the asynchronous schedule. See Section Asynchronous Schedule for complete operational details.

60.4.4.4.1 Example—Preserving Micro-Frame Integrity

One of the requirements of a USB host controller is to maintain *Frame Integrity.* This means that the HC must preserve the micro-frame boundaries. For example: SOF packets must be generated on time (within the specified allowable jitter), and High-speed EOF1,2 thresholds must be enforced. The end of micro-frame timing points EOF1 and EOF2 are clearly defined in the USB Specification Revision 2.0. One implication of this responsibility is that the HC must ensure that it does not start transactions that will not be completed before the end of the micro-frame. More precisely, no transactions should be started by the host controller, which cannot be completed in their entirety before the EOF1 point. In order to enforce this rule, the host controller must check each transaction before it starts to ensure that it will complete before the end of the micro-frame.

So, what exactly needs to be involved in this check? Fundamentally, the transaction data payload, plus bit stuffing, plus transaction overhead must be taken into consideration. It is possible to be extremely accurate on how much time the next transaction will take. Take OUTs for an example. The host controller must fetch all of the OUT data from memory in order to send it onto the USB bus. A host controller implementation could pre-fetch *all* of the OUT data, and pre-compute the actual number of bits in the token and data packets. In addition, the system knows the depth of the target endpoint, so it could closely estimate turnaround time for handshake. In addition, the host controller knows the size of a handshake packet. Pre-computing effects of bit stuffing and summing up the other overhead numbers could allow the host controller to know exactly whether there was enough bus time, before EOF1 to complete the OUT transaction. To accomplish this particular approach takes an inordinate amount of time and hardware complexity.

The alternative is to make a reasonable guess whether the next transaction can be started. An example approximation algorithm is described below. This example algorithm relies on the EHCI policy that periodic transactions are scheduled first in the micro-frame. It is a reasonable assumption that software will never over-commit the micro-frame to periodic transactions greater than the specification allowable 80%. In the available remaining 20% bandwidth, the host controller has some ability (in this example) to decide whether or not to execute a transaction. The result of this algorithm is that sometimes, under some circumstances a transaction will not be executed that *could* have been executed. However, under all circumstances, a transaction will never be started unless there is enough time in the frame to complete the transaction.

Transaction Fit—A Best-Fit Approximation Algorithm

A curve is calculated which represents the *latest* start time for every packet size, at which software will schedule the start of a periodic transaction. This curve is the 80% bandwidth curve. Another curve is calculated which is the absolute, latest permitted start time for every packet size. This curve represents the absolute latest time, that a transaction of each packet size can be started and completed, in the micro-frame. A plot of these two curves is illustrated in Figure 60-69. The plot Y-axis represents the number of byte-times left in a frame.

The space between the *80%* and the *Last Start* plots is bandwidth reclamation area. In this algorithm the host controller may skip transactions during this time if it is prudent.

The Best-Fit Approximation method plots a function ($f(x)$) between the 80% and *Last Start* curves. The function $f(x)$ adds a constant to every transaction's maximum packet size and the result compared with the number of bytes left in the frame. The constant represents an approximation of the effects of bit stuffing and protocol overhead. The host controller starts transactions whose results land *above* the function curve. The host controller will not start transactions whose results land below the function curve.

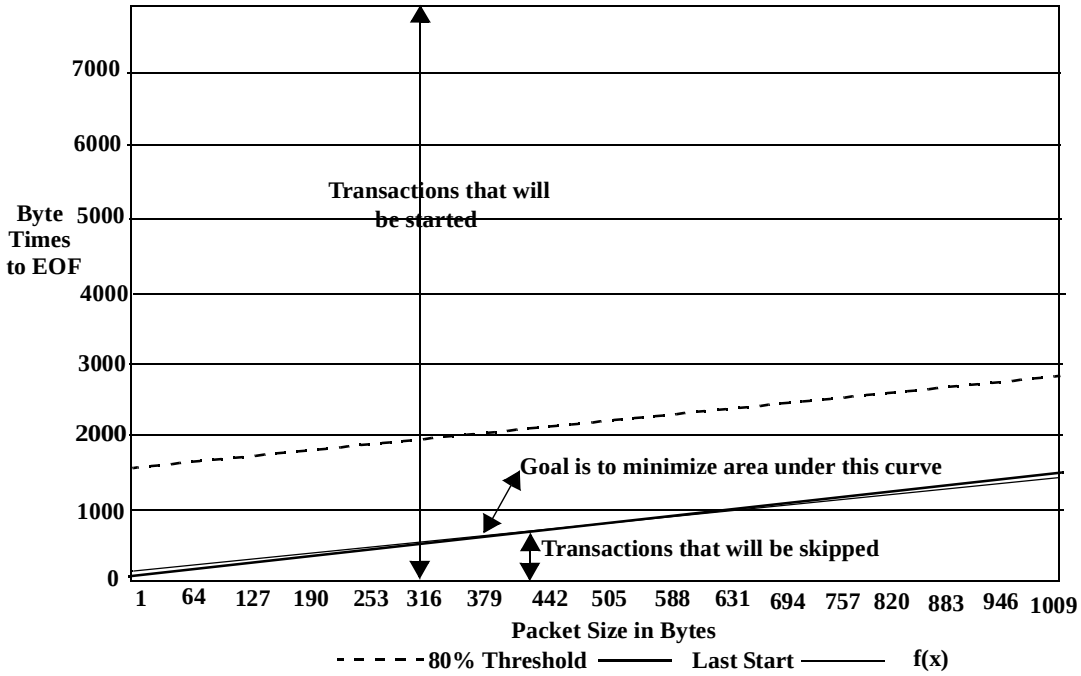


Figure 60-69. Best Fit Approximation

The *LastStart* line was calculated in this example to assume the absolute worst-case bus overhead per transaction. The particular transaction used was a start-split, zero-length OUT transaction with a handshake. Summaries of the component parts are listed in Table 60-91. The component times were derived from the protocol timings defined in the USB Specification Revision 2.0.

Table 60-91. Example Worse-case Transaction Timing Components

Component	Bit time	Byte Time	Explanation
Split Token	76	9.5	Split token as defined in USB core specification. Includes sync, token, eop, etc.
Host 2 Host IPG	88	11	Number of bit times required between consecutive host packets.
Token	67	8.375	Token as defined in USB core specification. Includessync, token, eop, etc.
Host 2 Host IPG	88	11	Same as above

Table 60-91. Example Worse-case Transaction Timing Components (continued)

Data Packet (0 data bytes)	66.7	8.34	Zero-length data packet. Includes sync, PID, crc16, eop, etc.
Turnaround time	721	90.125	Time for packet initiator (Host) to see the beginning of a response to a transmitted packet.
Handshake packet	48	6	Handshake packet as defined in USB core specification. Includes sync, PID, eop, etc.
		144	Total

The exact details of the function (f(x)) are up to the particular implementation. However, it should be obvious that the goal is to minimize the area under the curve between the approximation function and the *Last Start* curve, without dipping below the *LastStart* line, while at the same time keeping the check as simple as possible for hardware implementation. The f(x) in Figure 60-69 was constructed using the following pseudo-code test on each transaction size data point. This algorithm assumes that the host controller keeps track of the remaining bits in the frame.

```
Alorithm CheckTransactionWillFit (MaximumPacketSize, HC_BytesLeftInFrame)

Begin

        Local Temp = MaximumPacketSize + 192

        Local rvalue = TRUE


        If MaximumPacketSize >= 128 then

                Temp += 128

        End If


        If Temp > HC_BytesLeftInFrame then

                Rvalue = FALSE

        End If

        Return rvalue

End
```

This algorithm takes two inputs, the current maximum packet size of the transaction and a hardware counter of the number of bytes left in the current micro-frame. It unconditionally adds a simple constant of 192 to the maximum packet size to account for a first-order effect of transaction overhead and bit stuffing. If the transaction size is greater than or equal to 128 bytes, then an additional constant of 128 is added to the running sum to account for the additional worst-case bit stuffing of payloads larger than 128. An inflection point was inserted at 128 because the *f*(*x*) plot was getting *close* to the *LastStart* line.

60.4.4.5 Periodic Schedule Frame Boundaries vs Bus Frame Boundaries

The USB Specification Revision 2.0 requires that the frame boundaries (SOF frame number changes) of the high-speed bus and the full- and low-speed bus(s) below USB 2.0 Hubs be strictly aligned. Super-imposed on this requirement is that USB 2.0 Hubs manage full- and low-speed transactions via a micro-frame pipeline (see start- (SS) and complete- (CS) splits illustrated in Figure 60-70). A simple, direct projection of the frame boundary model into the host controller interface schedule architecture creates tension (complexity for both hardware and software) between the frame boundaries and the scheduling mechanisms required to service the full- and low-speed transaction translator periodic pipelines.

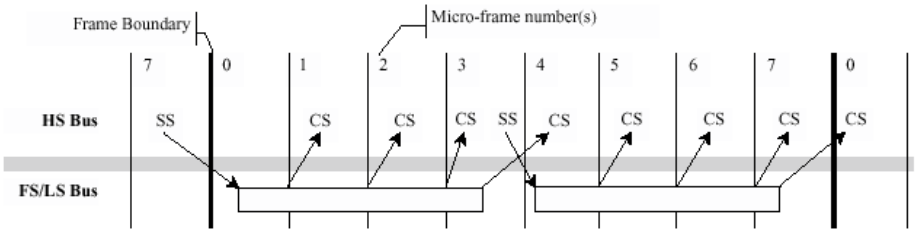


Figure 60-70. Frame Boundary Relationship between HS bus and FS/LS Bus

The simple projection, as Figure 60-70 illustrates, introduces frame-boundary wrap conditions for scheduling on both the beginning and end of a frame. In order to reduce the complexity for hardware and software, the host controller is required to implement a one micro-frame phase shift for its view of frame boundaries. The phase shift eliminates the beginning of frame and frame-wrap scheduling boundary conditions.

The implementation of this phase shift requires that the host controller use one register value for accessing the periodic frame list and another value for the frame number value included in the SOF token. These two values are separate, but tightly coupled. The periodic frame list is accessed via the Frame List Index Register (FRINDEX) documented in Section FRINDEX and initially illustrated in Section Schedule Traversal Rules . Bits FRINDEX[2:0], represent the micro-frame number. The SOF value is coupled to the value of FRINDEX[13:3]. Both FRINDEX[13:3] and the SOF value are increment based on FRINDEX[2:0]. It is required that the SOF value be delayed from the FRINDEX value by one micro-frame. The one micro-frame delay yields host controller periodic schedule and bus frame boundary relationship as illustrated in Figure 60-71. This adjustment allows software to trivially schedule the periodic start and complete-split transactions for full-and low-speed periodic endpoints, using the natural alignment of the periodic schedule interface. The reasons for selecting this phase-shift are beyond the scope of this specification.

Figure 60-71 illustrates how periodic schedule data structures relate to schedule frame boundaries and bus frame boundaries. To aid the presentation, two terms are defined. The host controller's view of the

1-millisecond boundaries is called *H-Frames*. The high-speed bus's view of the 1-millisecond boundaries is called *B-Frames.*

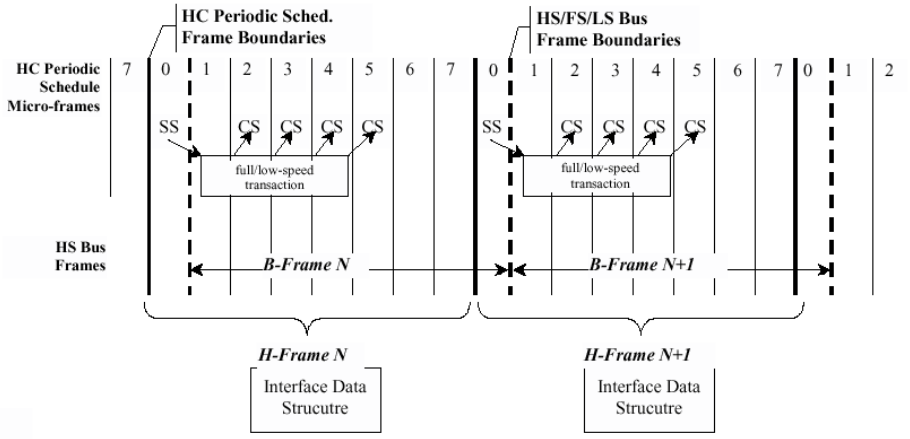


Figure 60-71. Relationship of Periodic Schedule Frame Boundaries to Bus Frame Boundaries

H-Frame boundaries for the host controller correspond to increments of FRINDEX[13:3]. Micro-frame numbers for the *H-Frame* are tracked by FRINDEX[2:0]. *B-Frame* boundaries are visible on the high-speed bus via changes in the SOF token's frame number. Micro-frame numbers on the high-speed bus are only derived from the SOF token's frame number (i.e. the high-speed bus will see eight SOFs with the same frame number value). *H-Frames* and *B-Frames* have the fixed relationship (i.e. *B-Frames* lag *H-Frames* by one micro-frame time) illustrated in Figure 60-71. The host controller's periodic schedule is naturally aligned to *H-Frames*. Software schedules transactions for full- and low-speed periodic endpoints relative the *H-Frames*. The result is these transactions execute on the high-speed bus at exactly the right time for the USB 2.0 Hub periodic pipeline. As described in Section FRINDEX, the SOF Value can be implemented as a shadow register (in this example, called SOFV), which lags the FRINDEX register bits [13:3] by one micro-frame count.

Table 60-92 illustrates the required relationship between the value of FRINDEX and the value of SOFV. This lag behavior can be accomplished by incrementing FRINDEX[13:3] based on carry-out on the 7 to 0 increment of FRINDEX[2:0] and incrementing SOFV based on the transition of 0 to 1 of FRINDEX[2:0].

Software is allowed to write to FRINDEX. Section FRINDEX provides the requirements that software should adhere when writing a new value in FRINDEX.

Table 60-92. Operation of FRINDEX and SOFV (SOF Value Register)

	Current			Next	
FRINDEX[F]	SOFV	FRINDEX[mF]	FRINDEX[F]	SOFV	FRINDEX[mF]
N	N	111b	N+1	N	000b
N+1	N	000b	N+1	N+1	001b
N+1	N+1	001b	N+1	N+1	010b
N+1	N+1	010b	N+1	N+1	011b
N+1	N+1	011b	N+1	N+1	100b

Table 60-92. Operation of FRINDEX and SOFV (SOF Value Register)

N+1	N+1	100b	N+1	N+1	101b
N+1	N+1	101b	N+1	N+1	110b
N+1	N+1	110b	N+1	N+1	111b

Where [F] = [13:3]; [µF] = [2:0]

60.4.4.6 Periodic Schedule

The periodic schedule traversal is enabled or disabled via the *Periodic Schedule Enable* bit in the USBCMD register. If the *Periodic Schedule Enable* bit is set to a zero, then the host controller simply does not try to access the periodic frame list via the *PERIODICLISTBASE* register. Likewise, when the *Periodic Schedule Enable* bit is a one, then the host controller does use the *PERIODICLISTBASE* register to traverse the periodic schedule. The host controller will not react to modifications to the *Periodic Schedule Enable* immediately. In order to eliminate conflicts with split transactions, the host controller evaluates the *Periodic Schedule Enable* bit only when FRINDEX[2:0] is zero. System software must not disable the periodic schedule if the schedule contains an active split transaction work item that spans the 000b micro-frame. These work items must be removed from the schedule before the *Periodic Schedule Enable* bit is written to a zero. The *Periodic Schedule Status* bit in the USBSTS register indicates status of the periodic schedule. System software enables (or disables) the periodic schedule by writing a one (or zero) to the *Periodic Schedule Enable* bit in the USBCMD register. Software then can poll the *Periodic Schedule Status* bit to determine when the periodic schedule has made the desired transition. Software must not modify the *Periodic Schedule Enable* bit unless the value of the *Periodic Schedule Enable* bit equals that of the *Periodic Schedule Status* bit.

The periodic schedule is used to manage all isochronous and interrupt transfer streams. The base of the periodic schedule is the periodic frame list. Software links schedule data structures to the periodic frame list to produce a graph of scheduled data structures. The graph represents an appropriate sequence of transactions on the USB. Figure 60-72 illustrates isochronous transfers (using iTDs and siTDs) with a period of one are linked directly to the periodic frame list. Interrupt transfers (are managed with queue heads) and isochronous streams with periods other than one are linked following the period-one iTD/siTDs. Interrupt queue heads are linked into the frame list ordered by poll rate. Longer poll rates are

linked first (for example, closest to the periodic frame list), followed by shorter poll rates, with queue heads with a poll rate of one, on the very end.

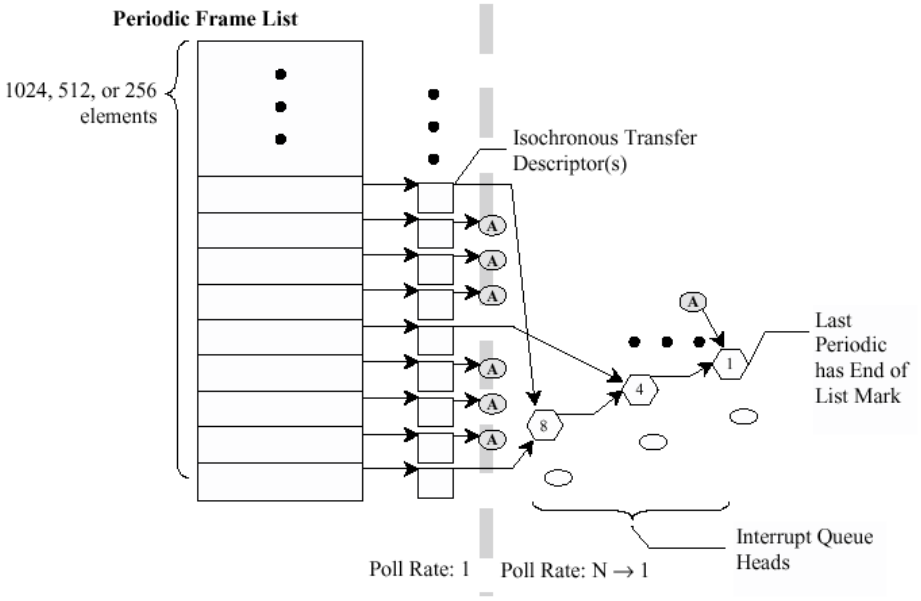


Figure 60-72. Example Periodic Schedule

60.4.4.7 Managing Isochronous Transfers Using iTDs

The structure of an iTD is presented in Isochronous (High-Speed) Transfer Descriptor (iTD). There are four distinct sections to an iTD:

- The first field is the *Next Link Pointer.* This field is for schedule linkage purposes only;
- Transaction description array. This area is an eight-element array. Each element represents control and status information for one micro-frame's worth of transactions for a single high-speed isochronous endpoint.
- The buffer page pointer array is a 7-element array of physical memory pointers to data buffers. These are 4K aligned pointers to physical memory.
- Endpoint capabilities. This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size and high-bandwidth multiplier.

60.4.4.7.1 Host Controller Operational Model for iTDs

The host controller uses FRINDEX register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to FRINDEX register bits [2:0]. Each iTD can span 8 micro-frames worth of transactions. When the host controller fetches an iTD, it uses FRINDEX register bits [2:0] to index into the transaction description array. If the *active* bit in the *Status* field of the indexed transaction description is set to zero, the host controller ignores the iTD and follows the *Next* pointer to the next schedule data structure.

When the indexed *active* bit is a one the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, etc.). It also uses the *Page Select (PG)* field to index the *buffer pointer* array, storing the selected *buffer pointer* and the next sequential *buffer pointer*. For example, if *PG* field is a 0, then the host controller will store Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's *PG* field) and the transaction description's *Transaction Offset* field. The host controller uses the endpoint addressing information and *I/O-bit* to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the active bit and writes back any additional status information to the *Status* field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical memory address) is constructed by concatenating the page pointer (example: page 0 pointer) selected by the active transaction descriptions' *PG* (example value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer will cross a page boundary. When this occurs the host controller simply replaces the current buffer pointer's page portion with the next page pointer (example: page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the *Maximum Packet Size* field. An iTD supports high-bandwidth pipes via the *Mult* (multiplier) field. When the *Mult* field is 1, 2, or 3, the host controller executes the specified number of *Maximum Packet* sized bus transactions for the endpoint in the current micro-frame. In other words, the *Mult* field represents a transaction count for the endpoint in the current micro-frame. If the *Mult* field is zero, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the *Mult* field.

For OUT transfers, the value of the *Transaction X Length* field represents the total bytes to be sent during the micro-frame. The *Mult* field must be set by software to be consistent with *Transaction X Length* and *Maximum Packet Sixe.* The host controller will send the bytes in *Maximum Packet Size*'d portions. After each transaction, the host controller decrements it's local copy of *Transaction X Length* by *Maximum Packet Size*. The number of bytes the host controller sends is always *Maximum Packet Size* or *Transaction X Length*, whichever is less. The host controller advances the transfer state in the transfer description, updates the appropriate record in the iTD and moves to the next schedule data structure. The maximum sized transaction supported is 3×1024 bytes.

For IN transfers, the host controller issues *Mult* transactions. It is assumed that software has properly initialized the iTD to accommodate all of the possible data. During each IN transaction, the host controller must use *Maximum Packet Size* to detect packet babble errors. The host controller keeps the sum of bytes received in the *Transaction X Length* field. After all transactions for the endpoint have completed for the micro-frame, *Transaction X Length* contains the total bytes received. If the final value of *Transaction X Length* is less than the value of *Maximum Packet Size*, then less data than was allowed for was received from the associated endpoint. This *short packet* condition does not set the *USBINT* bit in the USBSTS register to a one. The host controller will not detect this condition. If the device sends more than *Transaction X Length* or *Maximum Packet Size* bytes (whichever is less), then the host controller will set the *Babble Detected* bit to a one and set the *Active* bit to a zero. Note, that the host controller is not required

to update the iTD field *Transaction X Length* in this error scenario. If the *Mult* field is greater than one, then the host controller will automatically execute the value of *Mult* transactions. The host controller will not execute all *Mult* transactions if any of the following occurs:

- The endpoint is an OUT and *Transaction X Length* goes to zero before all the *Mult* transactions have executed (ran out of data), or
- The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before *Mult* transactions have been executed. The end of micro-frame may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next micro-frame. Refer to Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

60.4.4.7.2 Software Operational Model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N micro-frames. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

Figure 60-73 illustrates the simple model of how a client buffer is mapped by system software to the periodic schedule (i.e. the periodic frame list and a set of iTDs). On the right is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer should be used with each bus transaction. In the middle is the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one micro-frame's worth of transactions. The EHCI controller does not provide per-transaction results within a micro-frame. It treats the per-micro-frame transactions as a single logical transfer. On the left is the host controller’s frame list. System software establishes references from the appropriate locations in the frame list to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs

to service the entire buffer. System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.

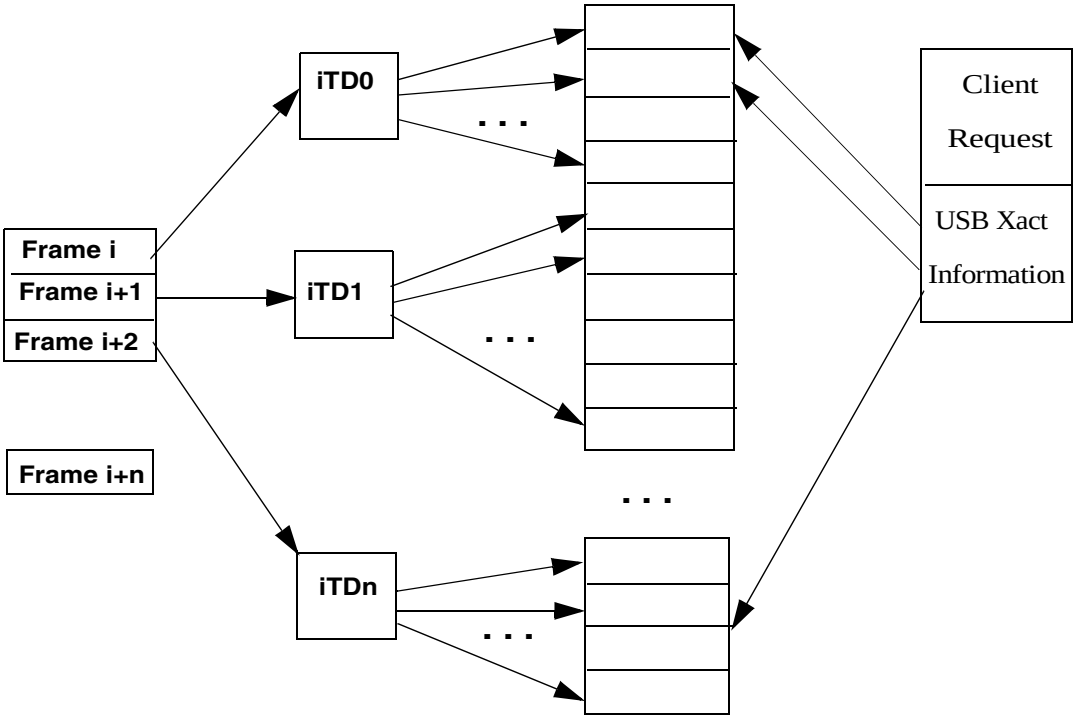


Figure 60-73. Example Association of iTDs to Client Request Buffer

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the *PG* field is set to index the correct physical buffer page pointer and the *Transaction Offset* field is set relative to the correct buffer pointer page (for example, the same one referenced by the *PG* field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current *Page Buffer Pointer* (as selected by the *PG* field) and concatenates to the *transaction offset* field. The result is a starting buffer address for the transaction. As the host controller moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available *Page Buffer Pointer*. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer will wrap a page boundary. Doing so will yield undefined behavior. The host controller hardware is not required to 'alias' the page selector to page zero. USB 2.0 isochronous endpoints can specify a period greater than one. Software can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the frame list) and by setting appropriate transaction description elements active bits to a one.

Periodic Scheduling Threshold

The *Isochronous Scheduling Threshold* field in the HCCPARAMS capability register is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures. It is used by system software when adding isochronous work items to the periodic schedule. The value of

this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

The iTD and siTD data structures each describe 8 micro-frames worth of transactions. The host controller is allowed to cache one (or more) of these data structures in order to reduce memory traffic. There are three basic caching models that account for the fact the isochronous data structures span 8 micro-frames. The three caching models are: no caching, micro-frame caching and frame caching.

When software is adding new isochronous transactions to the schedule, it always performs a read of the FRINDEX register to determine the current frame and micro-frame the host controller is currently executing. Of course, there is no information about where in the micro-frame the host controller is, so a constant uncertainty-factor of one micro-frame has to be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the *Isochronous Scheduling Threshold* field. The host controller may pre-fetch data structures during a periodic schedule traversal (per micro-frame) but will always dump any accumulated schedule state at the end of the micro-frame. At the appropriate time relative to the beginning of every micro-frame, the host controller always begins schedule traversal from the frame list. Software can use the value of the FRINDEX register (plus the constant 1 uncertainty-factor) to determine the approximate position of the executing host controller. When no caching is selected, software can add an isochronous transaction as near as 2 micro-frames in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the *Isochronous Scheduling Threshold* field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (8 micro-frames). Software uses the value of the FRINDEX register (plus the constant 1 uncertainty) to determine the *current* micro-frame/frame (assume modulo 8 arithmetic in adding the constant 1 to the micro-frame number). For any current frame N, if the current micro-frame is 0 to 6, then software can safely add isochronous transactions to Frame N + 1. If the current micro-frame is 7, then software can add isochronous transactions to Frame N + 2.

Micro-frame caching is indicated with a non-zero value in the least-significant 3 bits of the *Isochronous Scheduling Threshold* field. System software assumes the host controller caches one or more periodic data structures for the number of micro-frames indicated in the *Isochronous Scheduling Threshold* field. For example, if the count value were 2, then the host controller keeps a *window* of 2 micro-frames worth of state (current micro-frame, plus the next) on-chip. On each micro-frame boundary, the host controller releases the current micro-frame state and begins accumulating the next micro-frame state.

60.4.4.8 Asynchronous Schedule

The Asynchronous schedule traversal is enabled or disabled via the *Asynchronous Schedule Enable* bit in the USBCMD register. If the *Asynchronous Schedule Enable* bit is set to a zero, then the host controller simply does not try to access the asynchronous schedule via the *ASYNCLISTADDR* register. Likewise, when the *Asynchronous Schedule Enable* bit is a one, then the host controller does use the *ASYNCLISTADDR* register to traverse the asynchronous schedule. Modifications to the *Asynchronous Schedule Enable* bit are not necessarily immediate. Rather the new value of the bit will only be taken into

consideration the next time the host controller needs to use the value of the *ASYNCLISTADDR* register to get the next queue head.

The *Asynchronous Schedule Status* bit in the USBSTS register indicates status of the asynchronous schedule. System software enables (or disables) the asynchronous schedule by writing a one (or zero) to the *Asynchronous Schedule Enable* bit in the USBCMD register. Software then can poll the *Asynchronous Schedule Status* bit to determine when the asynchronous schedule has made the desired transition. Software must not modify the *Asynchronous Schedule Enable* bit unless the value of the *Asynchronous Schedule Enable* bit equals that of the *Asynchronous Schedule Status* bit.

The asynchronous schedule is used to manage all Control and Bulk transfers. Control and Bulk transfers are managed using queue head data structures. The asynchronous schedule is based at the *ASYNCLISTADDR* register. The default value of the *ASYNCLISTADDR* register after reset is undefined and the schedule is disabled when the *Asynchronous Schedule Enable* bit is a zero.

Software may only write this register with defined results when the schedule is disabled. for example, *Asynchronous Schedule Enable* bit in the USBCMD and the *Asynchronous Schedule Status* bit in the USBSTS register are zero. System software enables execution from the asynchronous schedule by writing a valid memory address (of a queue head) into this register. Then software enables the asychronous schedule by setting the *Asynchronous Schedule Enable* bit is set to one. The asynchronous schedule is actually enabled when the *Asynchronous Schedule Status* bit is a one.

When the host controller begins servicing the asynchronous schedule, it begins by using the value of the *ASYNCLISTADDR* register. It reads the first referenced data structure and begins executing transactions and traversing the linked list as appropriate. When the host controller "completes" processing the asynchronous schedule, it retains the value of the last accessed queue head's horizontal pointer in the *ASYNCLISTADDR* register. Next time the asynchronous schedule is accessed, this is the first data structure that will be serviced. This provides round-robin fairness for processing the asynchronous schedule.

A host controller "completes" processing the asynchronous schedule when one of the following events occur:

- The end of a micro-frame occurs.
- The host controller detects an empty list condition (i.e. see Section Empty Asynchronous Schedule Detection)
- The schedule has been disabled via the *Asynchronous Schedule Enable* bit in the USBCMD register.

The queue heads in the asynchronous list are linked into a simple circular list as shown in Figure 60-68. Queue head data structures are the only valid data structures that may be linked into the asynchronous schedule. An isochronous transfer descriptor (iTD or siTD) in the asynchronous schedule yields undefined results.

The maximum packet size field in a queue head is sized to accommodate the use of this data structure for all non-isochronous transfer types. The USB Specification, Revision 2.0 specifies the maximum packet sizes for all transfer types and transfer speeds. System software should always parameterize the queue head data structures according to the core specification requirements.

60.4.4.8.1 Adding Queue Heads to Asynchronous Schedule

This is a software requirement section. There are two independent events for adding queue heads to the asynchronous schedule. The first is the initial activation of the asynchronous list. The second is inserting a new queue head into an activated asynchronous list.

Activation of the list is simple. System software writes the physical memory address of a queue head into the *ASYNCLISTADDR* register, then enables the list by setting the *Asynchronous Schedule Enable* bit in the USBCMD register to a one.

When inserting a queue head into an active list, software must ensure that the schedule is always coherent from the host controllers' point of view. This means that the system software must ensure that all queue head pointer fields are valid. For example qTD pointers have *T-Bit*s set to a one or reference valid qTDs and the *Horizontal Pointer* references a valid queue head data structure. The following algorithm represents the functional requirements:

```
InsertQueueHead (pQHeadCurrent, pQueueHeadNew)

        --
        -- Requirement: all inputs must be properly initialized.
        --
        -- pQHeadCurrent is a pointer to a queue head that is
        -- already in the active list
        -- pQHeadNew is a pointer to the queue head to be added
        --
        -- This algorithm links a new queue head into a existing
        -- list
        --
        pQueueHeadNew.HorizontalPointer = pQueueHeadCurrent.HorizontalPointer


        pQueueHeadCurrent.HorizontalPointer = physicalAddressOf(pQueueHeadNew)


End InsertQueueHead
```

60.4.4.8.2 Removing Queue Heads from Asynchronous Schedule

This is a software requirement section. There are two independent events for removing queue heads from the asynchronous schedule. The first is shutting down (deactivating) the asynchronous list. The second is extracting a single queue head from an activated list. Software deactivates the asynchronous schedule by setting the *Asynchronous Schedule Enable* bit in the USBCMD register to a zero. Software can determine when the list is idle when the *Asynchronous Schedule Status* bit in the USBSTS register is a zero. The normal mode of operation is that software removes queue heads from the asynchronous schedule without shutting it down. Software must not remove an active queue head from the schedule. Software should first

deactivate all active qTDs, wait for the queue head to go inactive, then remove the queue head from the asynchronous list. Software removes a queue head from the asynchronous list via the following algorithm. As illustrated, the unlinking is quite easy. Software merely must ensure all of the link pointers reachable by the host controller are kept consistent.

```
UnlinkQueueHead (pQHeadPrevious, pQueueHeadToUnlink, pQHeadNext)

        --

        -- Requirement: all inputs must be properly initialized.

        --

        -- pQHeadPrevious is a pointer to a queue head that

        -- references the queue head to remove

        -- pQHeadToUnlink is a pointer to the queue head to be

        -- removed

        -- pQheadNext is a pointer to a queue head still in the

        -- schedule. Software provides this pointer with the

        -- following strict rules:

        --      if the host software is one queue head, then

        --      pQHeadNext must be the same as

        --      QueueheadToUnlink.HorizontalPointer. If the host

        --      software is unlinking a consecutive series of

        --      queue heads, QHeadNext must be set by software to

        --      the  queue head remaining in the schedule.

        --

        -- This algorithm unlinks a queue head from a circular list

        --

        pQueueHeadPrevious.HorizontalPointer = pQueueHeadToUnlink.HorizontalPointer
pQueueHeadToUnlink.HorizontalPointer = pQHeadNext

End UnlinkQueueHead
```

If software removes the queue head with the *H-bit* set to a one, it must select another queue head still linked into the schedule and set its *H-bit* to a one. This should be completed before removing the queue head. The requirement is that software keep one queue head in the asynchronous schedule, with its *H-bit* set to a one. At the point software has removed one or more queue heads from the asynchronous schedule, it is unknown whether the host controller has a cached pointer to them. Similarly, it is unknown how long the host controller might retain the cached information, as it is implementation dependent and may be affected by the actual dynamics of the schedule load. Therefore, once software has removed a queue head from the asynchronous list, it must retain the coherency of the queue head (link pointers, etc.). It cannot disturb the removed queue heads until it knows that the host controller does not have a local copy of a pointer to any of the removed data structures.

The method software uses to determine when it is safe to modify a removed queue head is to handshake with the host controller. The handshake mechanism allows software to remove items from the asynchronous schedule, then execute a simple, lightweight handshake that is used by software as a key that it can free (or reuse) the memory associated the data structures it has removed from the asynchronous schedule.

The handshake is implemented with three bits in the host controller. The first bit is a command bit (*Interrupt on Async Advance Doorbell* bit in the USBCMD register) that allows software to inform the host controller that something has been removed from its asynchronous schedule. The second bit is a status bit (*Interrupt on Async Advance* bit in the USBSTS register) that the host controller sets after it has released all on-chip state that may potentially reference one of the data structures just removed. When the host controller sets this status bit to a one, it also sets the command bit to a zero. The third bit is an interrupt enable (*Interrupt on Async Advance* bit in the USBINTR register) that is matched with the status bit. If the status bit is a one and the interrupt enable bit is a one, then the host controller will assert a hardware interrupt.

Figure 60-74 illustrates a general example. In this example, consecutive queue heads (B and C) are unlinked from the schedule using the algorithm above. Before the unlink operation, the host controller has a copy of queue head A.

The unlink algorithm requires that as software unlinks each queue head, the unlinked queue head is loaded with the address of a queue head that will remain in the asynchronous schedule.

When the host controller observes that doorbell bit being set to a one, it makes a note of the local reachable schedule information. In this example, the local reachable schedule information includes both queue heads (A and B). It is sufficient that the host controller can set the status bit (and clear the doorbell bit) as soon as it has traversed beyond current reachable schedule information (i.e. traversed beyond queue head (B) in this example).

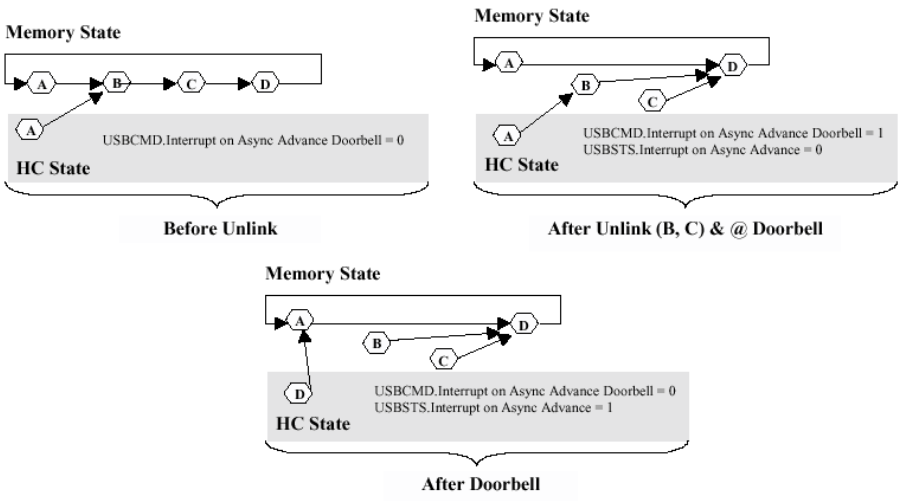


Figure 60-74. Generic Queue Head Unlink Scenario

Alternatively, a host controller implementation is allowed to traverse the entire asynchronous schedule list (for example, observed the head of the queue (twice)) before setting the *Advance on Async* status bit to a one.

Software may re-use the memory associated with the removed queue heads after it observes the *Interrupt on Async Advance* status bit is set to a one, following assertion of the doorbell. Software should acknowledge the *Interrupt on Async Advance* status as indicated in the USBSTS register, before using the doorbell handshake again.

60.4.4.8.3 Empty Asynchronous Schedule Detection

The Enhanced Host Controller Interface uses two bits to detect when the asynchronous schedule is empty. The queue head data structure (see Figure 60-63) defines an *H-bit* in the queue head, which allows software to mark a queue head as being the *head* of the reclaim list. The Enhanced Host Controller Interface also keeps a 1-bit flag in the USBSTS register (*Reclamation*) that is set to a zero when the Enhanced Interface Host Controller observes a queue head with the H-bit set to a one. The reclamation flag in the status register is set to one when any USB transaction from the asynchronous schedule is executed (or whenever the asynchronous schedule starts, see Section Asynchronous Schedule Traversal: Start Event).

If the Enhanced Host Controller Interface ever encounters an *H-bit* of one and a *Reclamation* bit of zero, the EHCI controller simply stops traversal of the asynchronous schedule.

An example illustrating the H-bit in a schedule is illustrated in Figure 60-75

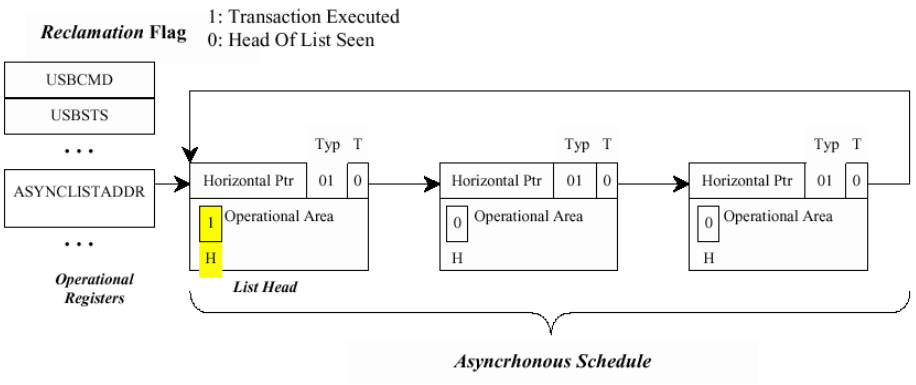


Figure 60-75. Asynchronous Schedule List with Annotation to Mark Head of List

Software must ensure there is at most one queue head with the *H-bit* set to a one, and that it is always coherent with respect to the schedule.

60.4.4.8.4 Restarting Asynchronous Schedule Before EOF

There are many situations where the host controller will detect an empty list *long* before the end of the micro-frame. It is important to remember that under many circumstances the schedule traversal has stopped due to Nak/Nyet responses from all endpoints.

An example of particular interest is when a start-split for a bulk endpoint occurs early in the micro-frame. Given the EHCI simple traversal rules, the complete-split for that transaction may Nak/Nyet out very quickly. If it is the only item in the schedule, then the host controller will cease traversal of the Asynchronous schedule very early in the micro-frame. In order to provide reasonable service to this endpoint, the host controller should issue the complete-split before the end of the current micro-frame, instead of waiting until the next micro-frame. When the reason for host controller idling asynchronous

schedule traversal is because of empty list detection, it is mandatory the host controller implement a 'waking' method to resume traversal of the asynchronous schedule. An example method is described below.

Example Method for Restarting Asynchronous Schedule Traversal

The reason for idling the host controller when the list is empty is to keep the host controller from unnecessarily occupying too much memory bandwidth. The question is: *how long should the host controller stay idle before restarting*?

The answer in this example is based on deriving a manifest constant, which is the amount of time the host controller will stay idle before restarting traversal. In this example, the manifest constant is called *AsyncSchedSleepTime*, and has a value of 10µsec. The value is derived based on the analysis in Section Section , Example Derivation for AsyncSchedSleepTime," The traversal algorithm is as follows:

- Traverse the Asynchronous schedule until the either an End-Of-micro-Frame event occurs, or an empty list is detected. If the event is an End-of-micro-Frame, go attempt to traverse the Periodic schedule. If the event is an empty list, then set a sleep timer and go to a *schedule sleep* state.
- When the sleep timer expires, set working context to the Asynchronous Schedule start condition and go to *schedule active* state. The start context allows the HC to reload *Nakcnt* fields, etc. so the HC has a chance to run for more than one iteration through the schedule.

This process simply repeats itself each micro-frame. Figure 60-76 illustrates a sample state machine to manage the active and sleep states of the Asynchronous Schedule traversal policy. There are three states: Actively traversing the Asynchronous schedule, Sleeping, and Not Active. The last two are similar in terms of interaction with the Asynchronous schedule, but the Not Active state means that the host controller is busy with the Periodic schedule or the Asynchronous schedule is not enabled. The Sleeping state is specifically a special state where the host controller is just waiting for a period of time before resuming execution of the Asynchronous schedule.

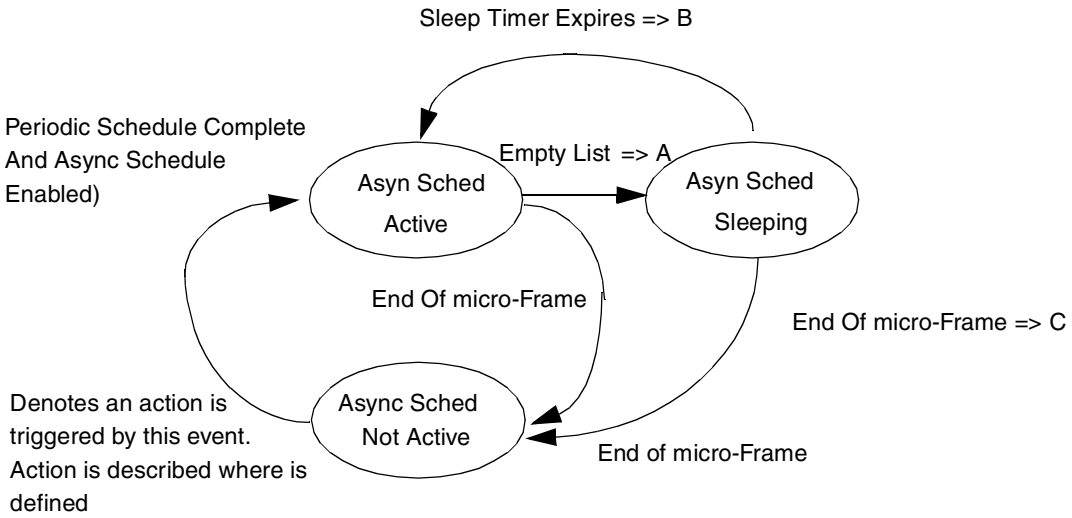


Figure 60-76. Example State Machine for Managing Asynchronous Schedule Traversal

The actions referred to in Figure 60-76 are defined in Table 60-93.

Table 60-93. Asynchronous Schedule SM Transition Actions

Action	Action Description Label
A	On detection of the empty list, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.
B	When the *AsynchronousTraversalSleepTimer* expires, the host controller sets the *Reclamation* bit in the USBSTS register to a one and moves the Nak Counter reload state machine to WaitForListHead (see Section Nak Count Reload Control).
C	The host controller cancels the sleep timer (*AsynchronousTraversalSleepTimer*).

Async Sched Not Active

This is the initial state of the traversal state machine after a host controller reset. The traversal state machine will not leave this state when the *Asynchronous Schedule Enable* bit in the USBCMD register is a zero.

This state is entered from **Async Sched Active** or **Async Sched Sleeping** states when the end-of-micro-frame event is detected.

Async Sched Active

This state is entered from the **Async Sched Not Active** state when the periodic schedule is not active. It is also entered from the **Async Sched Sleeping** states when the *AsyncrhonousTraversalSleepTimer* expires. On every transition into this state, the host controller sets the *Reclamation* bit in the USBSTS register to a one.

While in this state, the host controller will continually traverse the asynchronous schedule until either the end of micro-frame or an empty list condition is detected.

Async Sched Sleeping

The state is entered from the **Async Sched Active** state when a schedule empty condition is detected. On entry to this state, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.

Example Derivation for *AsyncSchedSleepTime*

The derivation is based on analysis of what work the host controller could be doing next. It assumes the host controller does not keep any state about what work is possibly pending in the asynchronous schedule. The schedule could contain any mix of the possible combinations of high- full- or low-speed control and bulk requests. Table 60-94 summarizes some of the typical 'next transactions' that could be in the schedule, and the amount of time (for example, *footprint*, or *wall clock*) the transaction will take to complete.

Table 60-94. Typical Low-/Full-speed Transaction Times

Transaction Attributes		Footprint (time)	Description
Speed	HS	11.9 ms	Maximum foot print for a worst-case, full-sized bulk data transaction.

Table 60-94. Typical Low-/Full-speed Transaction Times

Size	512	9.45 ms	Maximum footprint for an approximate best-case, full-sized bulk data transaction.
Type	Bulk		
Speed	FS	~50 ms	Approximate typical for full-sized bulk data. An 8-byte low-speed is about 2x, or between 90 and 100 ms.
Size	64		
Type	Bulk		
Speed	FS	~12 ms	Approximate typical for 8-byte bulk/control (i.e. setup)
Size	8		
Type	Cntrl		

A *AsyncSchedSleepTime* value of 10 µs provides a reasonable relaxation of the system memory load and still provides a good level of service for the various transfer types and payload sizes. For example, say we detect an empty list after issuing a start-split for a 64-byte full-speed bulk request. Assuming this is the only thing in the list, the host controller will get the results of the full-speed transaction from the hub during the fifth complete-split request. If the full-speed transaction was an IN and it nak'd, the 10µs sleep period would allow the host controller to get the NAK results on the first complete-split.

60.4.4.8.5 Asynchronous Schedule Traversal: *Start Event*

Once the HC has *idled* itself via the empty schedule detection (Section 0), it will naturally *activate* and begin processing from the Periodic Schedule at the beginning of each micro-frame. In addition, it may have idled itself early in a micro-frame. When this occurs (idles early in the micro-frame) the HC must occasionally *re-activate* during the micro-frame and traverse the asynchronous schedule to determine whether any progress can be made. The requirements and method for this restart are described in Section Restarting Asynchronous Schedule Before EOF . Asynchronous schedule *Start Event*s are defined to be:

- Whenever the host controller transitions from the periodic schedule to the asynchronous schedule. If the periodic schedule is disabled and the asynchronous schedule is enabled, then the beginning of the micro-frame is equivalent to the transition from the periodic schedule, or
- The asynchronous schedule traversal restarts from a sleeping state (see Section Restarting Asynchronous Schedule Before EOF).

60.4.4.8.6 Reclamation Status Bit (USBSTS Register)

The operation of the empty asynchronous schedule detection feature (Section Empty Asynchronous Schedule Detection) depends on the proper management of the *Reclamation* bit in the USBSTS register. The host controller tests for an empty schedule just after it fetches a new queue head while traversing the asynchronous schedule (See Section Fetch Queue Head).

It is required that the host controller sets the *Reclamation* bit to a one whenever an asynchronous schedule traversal *Start Event*, as documented in Section Asynchronous Schedule Traversal: Start Event , occurs. The *Reclamation* bit is also set to a one whenever the host controller executes a transaction while traversing the asynchronous schedule (see Section Execute Transaction). The host controller sets the

Reclamation bit to a zero whenever it finds a queue head with its *H-bit* set to a one. Software should only set a queue head's *H-bit* if the queue head is in the asynchronous schedule. If software sets the *H-bit* in an interrupt queue head to a one, the resulting behavior is undefined. The host controller may set the *Reclamation* bit to a zero when executing from the periodic schedule.

60.4.4.9 Operational Model for Nak Counter

This section describes the operational model for the *NakCnt* field defined in a queue head (see Section Queue Head Initialization). Software should not use this feature for interrupt queue heads. This rule is not required to be enforced by the host controller.

USB protocol has built-in flow control via the Nak response by a device. There are several scenarios, beyond the Ping feature, where an endpoint may naturally Nak or Nyet the majority of the time. An example is the host controller management of the split transaction protocol for control and bulk endpoints. All bulk endpoints (High- or Full-speed) are serviced via the same asynchronous schedule. The time between the *Start-split* transaction and the first *Complete-split* transaction could be very short (i.e. like when the endpoint is the only one in the asynchronous schedule). The hub NYETs (effectively Naks) the *Complete-split* transaction until the classic transaction is complete. This could result in the host controller thrashing memory, repeatedly fetching the queue head and executing the transaction to the Hub, which will not complete until after the transaction on the classic bus completes.

There are two component fields in a queue head to support the throttling feature: a counter field (*NakCnt*), and a counter reload field (*RL*). *NakCnt* is used by the host controller as one of the criteria to determine whether or not to execute a transaction to the endpoint. There are two operational modes associated with this counter:

- **Not Used.** This mode is set when the *RL* field is zero. The host controller ignores the *NakCnt* field for any execution of transactions through a queue head with an *RL* field of zero. Software must use this selection for interrupt endpoints.
- **Nak Throttle Mode.** This mode is selected when the *RL* field is non-zero. In this mode, the value in the *NakCnt* field represents the maximum number of Nak or Nyet responses the host controller will tolerate on each endpoint. In this mode, the HC will decrement the *NakCnt* field based on the token/handshake criteria listed in Table 60-95 The host controller must reload *NakCnt* when the endpoint successfully moves data (for example, policy to reward device for moving data).

Table 60-95. NakCnt Field Adjustment Rules

Token	Handshake	
	Handshake NAK	NYET
IN/PING	decrement *NakCnt*	N/A (protocol error)
OUT	decrement *NakCnt*	No Action[1]Start
Split	decrement *NakCnt*	N/A (protocol error)
Complete Split	No Action	Decrement *NakCnt*

[1] Recommended behavior on this response is to reload *NakCnt*

In summary, system software enables the counter by setting the reload field (*RL*) to a non-zero value. The host controller may execute a transaction if *NakCnt* is non-zero. The host controller will not execute a transaction if *NakCnt* is zero. The reload mechanism is described in detail in Section Nak Count Reload Control .

Note that when all queue heads in the Asynchronous Schedule either exhausts all transfers or all NakCnt's go to zero, then the host controller will detect an empty Asynchronous Schedule and idle schedule traversal (see Section Empty Asynchronous Schedule Detection).

Any time the host controller begins a new traversal of the Asynchronous Schedule, a *Start Event* is assumed, see Section Asynchronous Schedule Traversal: Start Event . Every time a Start-Event occurs, the Nak Count reload procedure is enabled.

60.4.4.9.1 Nak Count Reload Control

When the host controller reaches the *Execute Transaction* state for a queue head (meaning that it has an active operational state), it checks to determine whether the *NakCnt* field should be reloaded from *RL* (see Section Execute Transaction). If the answer is yes, then *RL* is copied into *NakCnt*. After the reload or if the reload is not active, the host controller evaluates whether to execute the transaction.

The host controller must reload nak counters (*NakCnt* see Figure 60-63) in queue heads during the first pass through the reclamation list after an asynchronous schedule Start Event (see Section Asynchronous Schedule Traversal: Start Event for the definition of the Start Event). The Asynchronous Schedule should have at most one queue head marked as the head (see Figure 60-75). Figure 60-77 illustrates an example state machine that satisfies the operational requirements of the host controller detecting the first pass through the Asynchronous Schedule. This state machine is maintained internal to the host controller and is only used to gate reloading of the nak counter during the queue head traversal state: **Execute Transaction** (Figure 60-77). The host controller does not perform the nak counter reload operation if the RL field (see Figure 60-63) is set to zero.

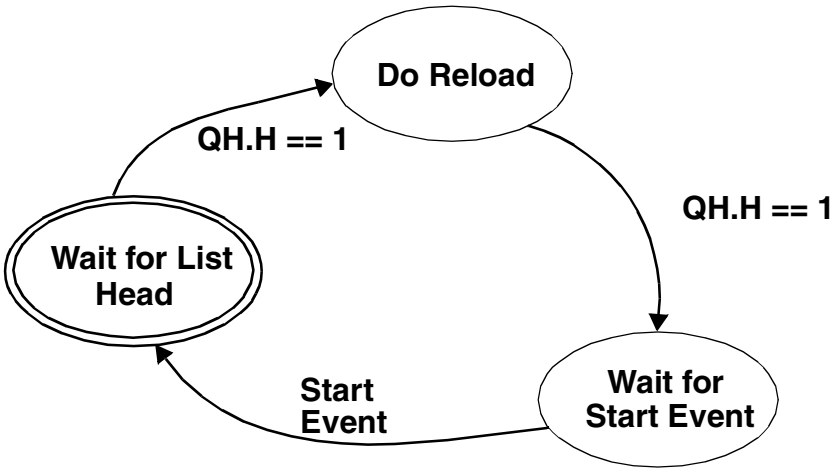


Figure 60-77. Example HC State Machine for Controlling Nak Counter Reloads

Wait for List Head

This is the initial state. The state machine enters this state from **Wait for Start Event** when a start event as defined in Section Asynchronous Schedule Traversal: Start Event occurs. The purpose of this state is to wait for the first observation of the head of the Asynchronous Schedule. This occurs when the host controller fetches a queue head whose *H-bit* is set to a one.

Do Reload

This state is entered from the **Wait for List Head** state when the host controller fetches a queue head with the *H-bit* set to a one. While in this state, the host controller will perform nak counter reloads for every queue head visited that has a non-zero nak reload value (*RL*) field.

Wait for Start Event

This state is entered from the *Do Reload* state when a queue head with the *H-bit* set to a one is fetched. While in this state, the host controller will not perform nak counter reloads.

60.4.4.10 Managing Control/Bulk/Interrupt Transfers via Queue Heads

This section presents an overview of how the host controller interacts with queuing data structures.

Queue heads use the Queue Element Transfer Descriptor (qTD) structure. One queue head is used to manage the data stream for one endpoint. The queue head structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see Overlay area defined in Figure 60-63). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a *transfer.*

The general processing model for the host controller's use of a queue head is simple:

- read a queue head,
- execute a transaction from the overlay area,
- write back the results of the transaction to the overlay area
- move to the next queue head.

If the host controller encounters errors during a transaction, the host controller will set one (or more) of the error reporting bits in the queue head's *Status* field. The *Status* field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the queue head *Status* field are 'sticky' until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so no additional bus transactions will occur for the endpoint and the host controller will not advance the queue.

An example host controller operational state machine of a queue head traversal is illustrated in Figure 60-78. This state machine is a model for how a host controller should traverse a queue head. The host controller must be able to advance the queue from the *Fetch QH* state in order to avoid all

hardware/software race conditions. This simple mechanism allows software to simply link qTDs to the queue head and *activate* them, then the host controller will always *find* them if/when they are reachable.

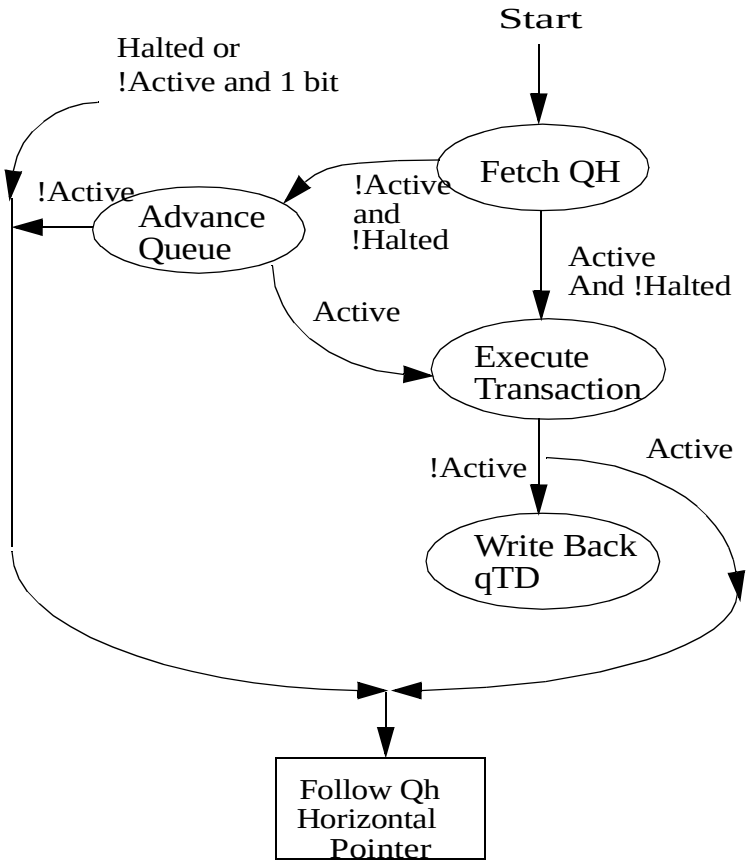


Figure 60-78. Host Controller Queue Head Traversal State Machine

This traversal state machine applies to all queue heads, regardless of transfer type or whether split transactions are required. The following sections describe each state. Each state description describes the entry criteria. The **Execute Transaction** state (Section Execute Transaction) describes the basic requirements for all endpoints. Sections Split Transactions for Asynchronous Transfers and Split Transaction Interrupt describe details of the required extensions to the **Execute Transaction** state for endpoints requiring split transactions.

Note: Prior to software placing a queue head into either the periodic or asynchronous list, software must ensure the queue head is properly initialized. Minimally, the queue head should be initialized to the following (see SectionQueue Head for layout of a queue head):

- Valid static endpoint state
- For the very first use of a queue head, software may zero-out the queue head transfer overlay, then set the *Next qTD Pointer* field value to reference a valid qTD.

60.4.4.10.1 Fetch Queue Head

A queue head can be referenced from the physical address stored in the ASYNCLISTADDR Register (Section ASYNCLISTADDR;ENDPOINTLISTADDR). Additionally, it may be referenced from the *Next Link Pointer* field of an iTD, siTD, FSTN or another Queue Head. If the referencing link pointer has the *Typ* field set to indicate a queue head, it is assumed to reference a queue head structure as defined in Figure 60-63.

While in this state, the host controller performs operations to implement empty schedule detection (Section Empty Asynchronous Schedule Detection) and Nak Counter reloads (Section Operational Model for Nak Counter). After the queue head has been fetched, the host controller conducts the following queries for empty schedule detection:

- If queue head is not an interrupt queue head (i.e. *S-mask* is a zero), and
- The *H-bit* is a one, and
- The *Reclamation* bit in the USBSTS register is a zero.

When these criteria are met, the host controller will stop traversing the asynchronous list (as described in Section Empty Asynchronous Schedule Detection). When the criteria are not met, the host controller continues schedule traversal. If the queue head is not an interrupt and the *H-bit* is a one and the *Reclamation* bit is a one, then the host controller sets the *Reclamation* bit in the USBSTS register to a zero before completing this state. The operations for reloading of the Nak Counter are described in detail in Section Operational Model for Nak Counter.

This state is complete when the queue head has been read on-chip.

60.4.4.10.2 Advance Queue

To advance the queue, the host controller must find the next qTD, adjust pointers, perform the overlay and write back the results to the queue head.

This state is entered from the **FetchQHD** state if the overlay *Active* and *Halt* bits are set to zero. On entry to this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD and determines whether or not to perform an overlay. Note that if the *I-bit* is a one and the *Active* bit is a zero, the host controller immediately skips processing of this queue head, exits this state and uses the horizontal pointer to the next schedule data structure. If the field *Bytes to Transfer* is not zero and the *T-bit* in the *Alternate Next qTD Pointer* is set to zero, then the host controller uses the *Alternate Next qTD Pointer.* Otherwise, the host controller uses the *Next qTD Pointer.* If *Next qTD Pointer*'s *T-bit* is set to a one, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer the host controller fetches the referenced qTD. If the fetched qTD has it's *Active* bit set to a one, the host controller moves the pointer value used to reach the qTD (*Next* or *Alternate Next*) to the *Current qTD Pointer* field, then performs the overlay. If the fetched qTD has its *Active* bit set to a zero, the host controller aborts the queue advance and follows the queue head's horizontal pointer to the next schedule data structure. The host controller performs the overlay based on the following rules:

- The value of the data toggle (*dt*) field in the overlay area depends on the value of the *data toggle control* (*dtc*) bit (see Table 60-82).

- If the *EPS* field indicates the endpoint is a high-speed endpoint, the *Ping* state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- *C-prog-mask* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *Frame Tag* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *NakCnt* field in the overlay area is loaded from the *RL* field in the queue head's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the queue head.

60.4.4.10.3 Execute Transaction

The host controller enters this state from the **Fetch Queue Head** state only if the *Active* bit in *Status* field of the queue head is set to a one.

On entry to this state, the host controller executes a few pre-operations, then checks some pre-condition criteria before committing to executing a transaction for the queue head.

The pre-operations performed and the pre-condition criteria depend on whether the queue head is an interrupt endpoint. The host controller can determine that a queue head is an interrupt queue head when the queue head's *S-mask* field contains a non-zero value. It is the responsibility of software to ensure the *S-mask* field is appropriately initialized based on the transfer type. There are other criteria that must be met if the *EPS* field indicates that the endpoint is a low- or full-speed endpoint, see Sections Split Transactions for Asynchronous Transfers and Split Transaction Interrupt .

Interrupt Transfer Pre-condition Criteria

If the queue head is for an interrupt endpoint (for example, non-zero *S-mask* field), then the FRINDEX[2:0] field must identify a bit in the *S-mask* field that has a one in it. For example, an *S-mask* value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met then the host controller considers this queue head for a transaction**.**

Asynchronous Transfer Pre-operations and Pre-condition Criteria

If the queue head is not for an interrupt endpoint (for example, a zero *S-mask* field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria: The pre-operation is:

Checks the Nak counter reload state (Section Operational Model for Nak Counter). It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Whether or not the *NakCnt* field has been reloaded, the host controller checks the value of the *NakCnt* field in the queue head. If *NakCnt* is non-zero, or if the *Reload Nak Counter* field is zero, then the host controller considers this queue head for a transaction.

Transfer Type Independent Pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter *qHTransactionCounter* is loaded from the queue head's *Mult* field. A host controller implementation is allowed to ignore this for queue heads on the asynchronous list. It is mandatory for interrupt queue heads. Software should ensure that the *Mult* field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the micro-frame to complete this transaction (see Section Transaction Fit—A Best-Fit Approximation Algorithm for an example evaluation method). If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of *qHTransactionCounter* for an interrupt endpoint is zero, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets the *Reclamation* bit in the USBSTS register to a one and then begins executing one or more transactions using the endpoint information in the queue head. The host controller iterates *qHTransactionCounter* times in this state executing transactions. After each transaction is executed, *qHTransactionCounter* is decremented by one. The host controller will exit this state when one of the following events occurs:

- The *qHTransactionCounter* decrements to zero
- The endpoint responds to the transaction with any handshake other than an ACK

NOTE

Note that for a high-bandwidth interrupt OUT endpoint, the host controller may optionally immediately retry the transaction if it fails.

- The transaction experiences a transaction error
- The *Active* bit in the queue head goes to a zero
- There is not enough time in the micro-frame left to execute the next transaction (see Section , Transaction Fit—A Best-Fit Approximation Algorithm" for example method for implementing the frame boundary test).

The results of each transaction is recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance queue head's transfer state, the *Total Bytes to Transfer* field is decremented by the number of bytes moved in the transaction, the data toggle bit (*dt*) is toggled, the current page offset is advanced to the next appropriate value (for example, advanced by the number of bytes successfully moved), and the *C_Page* field is updated to the appropriate value (if necessary). See Section Buffer Pointer List Use for Data Streaming with qTDs .

Note that the *Total Bytes To Transfer* field may be zero when all the other criteria for executing a transaction are met. When this occurs, the host controller will execute a zero-length transaction to the endpoint. If the *PID_Code* field indicates an IN transaction and the device delivers data, the host controller will detect a packet babble condition, set the *babble* and *halted* bits in the *Status* field, set the *Active* bit to a zero, write back the results to the source qTD, then exit this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example, not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement *qHTransactionCounter*). It is recommended (but not required) the host controller continue executing transactions for this endpoint if the resultant value of *qHTransactionCounter* is greater than one.

If the response to the IN bus transaction is a Nak (or Nyet) and *RL* is non-zero, *NakCnt* is decremented by one. If *RL* is zero, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of *CErr* did not change). Software should set the *RL* field to zero if the queue head is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly *NakCnt*, the host controller writes back the results of the transaction to the queue head's overlay area in main memory.

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is *Maximum Packet Size*. The number of bytes moved during an OUT transaction is either *Maximum Packet Length* bytes or *Total Bytes to Transfer*, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The *CErr* field is decremented by one and the status field is updated to reflect the type of error observed. Transaction errors are summarized in Section Transaction Error .

The following events will cause the host controller to clear the *Active* bit in the queue head's overlay status field. When the *Active* bit transitions from a one to a zero, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the *Active* bit) determines the next state.

- *CErr* field decrements to zero. When this occurs the *Halted* bit is set to a one and *Active* is set to a zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the *Halted* bit is set to a one and the *Active* bit is set to a zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The *Total Bytes to Transfer* field is zero after the transaction completes. Note that for a zero length transaction, it was zero before the transaction was started. When this condition occurs, the *Active* bit is set to zero.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the *Maximum Packet Length*. When this occurs, the *Active* bit is set to zero and a short packet condition exists. The short-packet condition is detected during the **Advance Queue** state. Refer to Section Split Transactions for additional rules for managing low- and full-speed transactions.

With the exception of a NAK response (when *RL* field is zero), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a high-speed endpoint, the queue head information written back includes minimally the following fields: The *PID Code* field indicates an IN and the device sends more than the expected number of bytes (e.g.

Maximum Packet Length or *Total Bytes to Transfer* bytes, whichever is less) (e.g. a packet babble). This results in the host controller setting the *Halted* bit to a one.

- NakCnt, dt, Total Bytes to Transfer, C_Page, Status, CERR, and Current Offset

For a low- or full-speed device the queue head information written back also includes the fields:

- C-prog-mask, FrameTag and S-bytes.

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

Halting a Queue Head

A halted endpoint is defined only for the transfer types that are managed via queue heads (control, bulk and interrupt). The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a queue head) the Host Controller halts the queue head and set the USBERRINT status bit in the USBSTS register to a one. To halt the queue head, the *Active* bit is set to a zero and the *Halted* bit is set to a one. There may be other error status bits that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller will not advance the transfer state on a transaction that results in a *Halt* condition (e.g. no updates necessary for *Total Bytes to Transfer*, *C_Page*, *Current Offset*, and *dt*). The host controller must update *CErr* as appropriate. When a queue head is halted, the *USB Error Interrupt* bit in the USBSTS register is set to a one. If the *USB Error Interrupt Enable* bit in the USBINTR register is set to a one, a hardware interrupt is generated at the next interrupt threshold.

Asynchronous Schedule Park Mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a high-speed queue head in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule. This feature has no effect on queue heads or other data structures in the Periodic schedule. This feature is similar in intent as the *Mult* feature that is used in the Periodic schedule. Where-as the *Mult* feature is a characteristic that is tunable for each endpoint; park-mode is a policy that is applied to all high-speed queue heads in the asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in Section Execute Transaction apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same queue head before moving to the next queue head in the Asynchronous List. This feature should allow the host controller to attain better bus utilization for those devices that are capable of moving data at maximum rate, while at the same time providing a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the *Asynchronous Schedule Park Capability* bit in the HCCPARAMs register to a one. This information keys system software that the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USBCMD register are modifiable. System software enables the feature by writing a one to the *Asynchronous Schedule Park Mode Enable* bit.

When park-mode is not enabled (e.g. *Asynchronous Schedule Park Mode Enable* bit in the USBCMD register is a zero), the host controller must not execute more than one bus transaction per high-speed queue head, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a queue head whose *EPS* field indicates a Low/Full-speed device (i.e. only one bus transaction is allowed from each Low/Full-speed queue head per traversal of the asynchronous schedule). Park-mode may only be applied to queue heads in the Asynchronous schedule whose *EPS* field indicates that it is a high-speed device.

The host controller must apply park mode to queue heads whose *EPS* field indicates a high-speed endpoint. The maximum number of consecutive bus transactions a host controller may execute on a high-speed queue head is determined by the value in the *Asynchronous Schedule Park Mode Count* field in the USBCMD register. Software must not set *Asynchronous Schedule Park Mode Enable* bit to a one and also set *Asynchronous Schedule Park Mode Count* field to a zero. The resulting behavior is not defined. An example behavioral example describes the operational requirements for the host controller implementing park-mode. This feature does not affect how the host controller handles the bus transaction as defined in Section Execute Transaction . It only effects how many consecutive bus transactions for the current queue head can be executed. All boundary conditions, error detection and reporting applies as usual. This feature is similar in concept to the use of the *Mult* field for high-bandwidth Interrupt for queue heads in the Periodic Schedule.

The host controller effectively loads an internal down-counter *PM-Count* from *Asynchronous Schedule Park Mode Count* when *Asyncrhonous Schedule Park Mode Enable* bit is a one, and a high-speed queue head is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, *PM-Count* is decremented. The host controller may continue to execute bus transactions from the current queue head until *PM-Count* goes to zero, an error is detected, the buffer for the current transfer is exhausted or the endpoint responds with a flow-control or STALL handshake. Table 60-96 summarizes the responses that effect whether the host controller continues with another bus transaction for the current queue head.

Table 60-96. Actions for Park Mode, based on Endpoint Response and Residual Transfer State

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	
IN	DATA[0,1] w/Maximum Packet sized data	Not zero	Not Zero	Allowed to perform another bus transaction.[1,2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.
	DATA[0,1] w/short packet	Don't care	Don't care	Retire qTD and move to next QH.
	NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH.

Table 60-96. Actions for Park Mode, based on Endpoint Response and Residual Transfer State (continued)

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	
OUT	ACK	Not zero	Not Zero	Allowed to perform another bus transaction. [2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't' care	Move to next QH.
	NYET, NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not Zero	Not Zero	Allowed to perform another bus transaction. [2]
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

[1] Note, the host controller may continue to execute bus transactions from the current high-speed queue head (if PM-Count is not equal to zero), if a PID mismatch is detected (e.g. expected DATA1 and received DATA0, or visa-versa).

[2] Note, this specification does not require that the host controller execute another bus transaction when PM-Count is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

60.4.4.10.4 Write Back qTD

This state is entered from the **Execute Transaction** state when the *Active* bit is set to a zero. The source data for the write-back is the transfer results area of the queue head overlay area (see Table 60-96). The host controller uses the *Current qTD Pointer* field as the target address for the qTD. The queue head transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as: qTD retirement. The fields that must be written back to the source qTD include *Total Bytes to Transfer*, *Cerr*, and *Status*.

The duration of this state depends on when the qTD write-back has been committed.

60.4.4.10.5 Follow Queue Head Horizontal Pointer

The host controller must use the horizontal pointer in the queue head to the next schedule data structure when any of the following conditions exist:

- If the *Active* bit is a one on exit from the **Execute Transaction** state, or
- When the host controller exits the **Write Back qTD** state, or
- If the **Advance Queue** state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is a one on exit from the **Fetch QH** state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

60.4.4.10.6 Buffer Pointer List Use for Data Streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*. This means: if the buffer spans more than one physical page, it must obey the following rules (Figure 60-79 illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4K chunk beyond the first page, each buffer portion matches to a full 4K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20K bytes. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16Kbyte buffer with any starting buffer alignment.

The host controller uses the field *C_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction will span a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

Figure 60-79 illustrates a nominal example of how System software would initialize the buffer pointers list and the *C_Page* field for a transfer size of 16383 bytes. *C_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical page. *Current Offset* (the lower 12-bits of queue head Dword 7) holds

the offset in the page e.g. 2049 (e.g. 4096–2047). The remaining page pointers are set to reference the beginning of each subsequent 4K page.

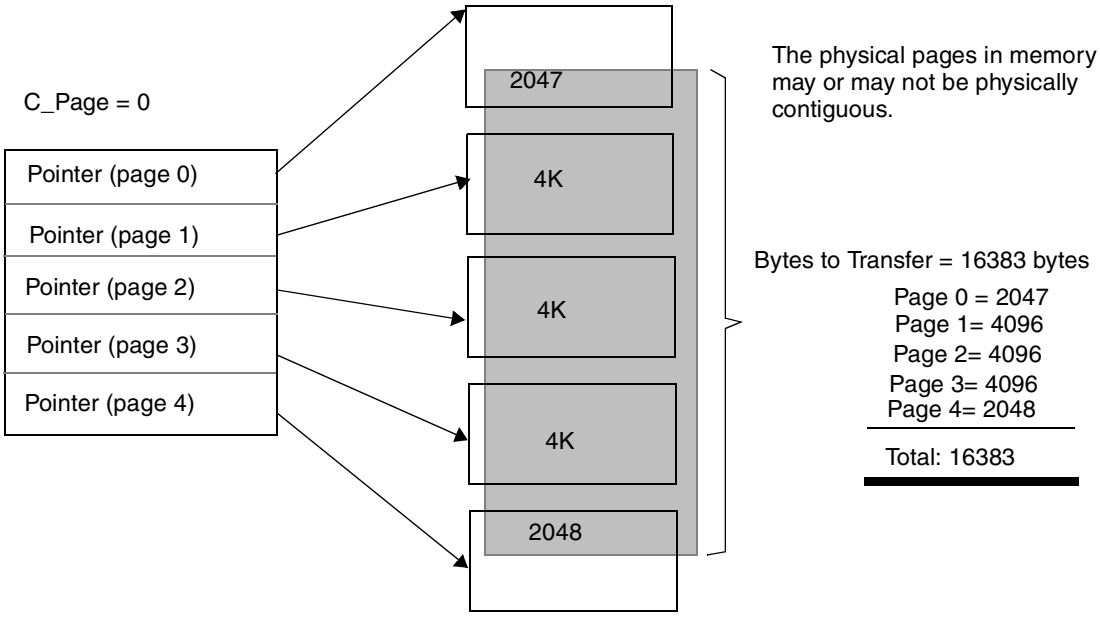


Figure 60-79. Example Mapping of qTD Buffer Pointers to Buffer Pages

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because *C_Page* is set to zero) and concatenates the *Current Offset* field. The 512 bytes are moved during the transaction, the *Current Offset* and *Total Bytes to Transfer* are adjusted by 512 and written back to the queue head working area.

During the 4th transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller will increment *C_Page* (to 1) and use the page 1 pointer to move the final byte of the transaction. After the 4th transaction, the active page pointer is the page 1 pointer and *Current Offset* has rolled to one, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (i.e. *C_Page*) when necessary. There are three conditions for how the host controller handles *C_Page*.

- The current transaction does not span a page boundary. The value of *C_Page* is not adjusted by the host controller.
- The current transaction does span a page boundary. The host controller must detect the page cross condition and advance to the next buffer while streaming data to/from the USB.
- The current transaction completes on a page boundary (i.e. the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment *C_Page* before writing back status for the transaction.

Note that the only valid adjustment the host controller may make to *C_Page* is to increment by one.

60.4.4.10.7 Adding Interrupt Queue Heads to the Periodic Schedule

The link path(s) from the periodic frame list to a queue head establishes in which frames a transaction can be executed for the queue head. Queue heads are linked into the periodic schedule so they are polled at the appropriate rate. System software sets a bit in a queue head's *S-Mask* to indicate which micro-frame with-in a 1 millisecond period a transaction should be executed for the queue head. Software must ensure that all queue heads in the periodic schedule have *S-Mask* set to a non-zero value. An *S-mask* with a zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of queue head linking and *S-Mask* values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in Table 60-97.

Table 60-97. Example Periodic Reference Patterns for Interrupt Transfers with 2ms Poll Rate

Frame # Reference Sequence	Description
0, 2, 4, 6, 8, etc. *S-Mask* = 01h	A queue head for the *bInterval* of 2 milliseconds (16 micro-frames) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the *S-Mask* field in the queue head is set to 01h, indicating that the transaction for the endpoint should be executed on the bus during micro-frame 0 of the frame.
0, 2, 4, 6, 8, etc. *S-Mask* = 02h	Another example of a queue head with a *bInterval* of 2 milliseconds is linked into the periodic frame list at exactly the same interval as the previous example. However, the *S-Mask* is set to 02h indicating that the transaction for the endpoint should be executed on the bus during micro-frame 1 of the frame.

60.4.4.10.8 Managing Transfer Complete Interrupts from Queue Heads

The host controller will set an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an *Interrupt on Complete* (*IOC*) bit set to a one, or whenever a transfer (qTD) completes with a short packet. If system software needs multiple qTDs to complete a client request (i.e. like a control transfer) the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOC's to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be re-used in a timely manner.

60.4.4.11 Ping Control

USB 2.0 defines an addition to the protocol for high-speed devices called Ping. Ping is required for all USB 2.0 High-speed bulk and control endpoints. Ping is not allowed for a split-transaction stream. This extension to the protocol eliminates the bad side-effects of Naking OUT endpoints. The *Status* field has a *Ping State* bit, which the host controller uses to determine the *next* actual PID it will use in the next transaction to the endpoint (see Table 60-79). The Ping State bit is only managed by the host controller for queue heads that meet the following criteria:

- Queue head is not an interrupt
- *EPS* field equals High-Speed
- *PIDCode* field equals OUT

Table 60-98 illustrates the state transition table for the host controller's responsibility for maintaining the PING protocol. Refer to Chapter 8 in the USB Specification Revision 2.0 for detailed description on the Ping protocol.

Table 60-98. Ping Control State Transition Table

	Event		
Current	**Host**	**Device**	**Next**
Do Ping	PING	Nak	Do Ping
Do Ping	PING	Ack	Do OUT
Do Ping	PING	XactErr[1]	Do Ping
Do Ping	PING	Stall	N/C[2] Do
OUT	OUT	Nak	Do Ping
Do OUT	OUT	Nyet	Do Ping
Do OUT	OUT	Ack	Do OUT
Do OUT	OUT	XactErr[1]	Do Ping
Do OUT	OUT	Stall	N/C[2]

1 Transaction Error (XactErr) is any time the host misses the handshake.

2 No transition change required for the Ping State bit. The Stall handshake results in the endpoint being halted (e.g. Active set to zero and Halt set to a one). Software intervention is required to restart queue. 3 A Nyet response to an OUT means that the device has accepted the data, but cannot receive any more at this time. Host must advance the transfer state and additionally, transition the Ping State bit to **Do Ping**. The Ping State bit has the following encoding:

Value	Meaning
0B	Do OUT The host controller will use an OUT PID during the next bus transaction to this endpoint.
1B	Do Ping The host controller will use a PING PID during the next bus transaction to this endpoint.

The defined ping protocol (see USB 2.0 Specification, Chapter 8) allows the host to be *imprecise* on the initialization of the ping protocol (i.e. start in *Do OUT* when we don't know whether there is space on the device or not). The host controller manages the *Ping State* bit. System software sets the initial value in the queue head when it initializes a queue head. The host controller preserves the *Ping State* bit across all queue advancements. This means that when a new qTD is written into the queue head overlay area, the previous value of the *Ping State* bit is preserved.

60.4.4.12 Split Transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs. This section describes how the host controller uses the interface data structures to manage data streams with full- and low-speed devices, connected below USB 2.0 hub, utilizing the split transaction protocol. Refer to USB 2.0 Specification for the complete definition of the split transaction protocol. Full- and Low-speed devices are enumerated identically as high-speed devices, but the transactions to the Full- and Low-speed endpoints use the split-transaction protocol on the high-speed bus. The split transaction protocol is an encapsulation of (or wrapper around) the Full- or Low-speed transaction. The high-speed

wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator below which the Full- or Low-speed device is attached.

The EHCI interface uses dedicated data structures for managing full-speed isochronous data streams (see Section Split Transaction Isochronous Transfer Descriptor (siTD)). Control, Bulk and Interrupt are managed using the queuing data structures (see Sections Queue Head). The interface data structures need to be programmed with the device address and the Transaction Translator number of the USB 2.0 Hub operating as the Low-/Full-speed host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

60.4.4.12.1 Split Transactions for Asynchronous Transfers

A queue head in the asynchronous schedule with an *EPS* field indicating a full-or low-speed device indicates to the host controller that it must use split transactions to stream data for this queue head. All full-speed bulk and full-, low-speed control are managed via queue heads in the asynchronous schedule.

Software must initialize the queue head with the appropriate device address and port number for the transaction translator that is serving as the full/low-speed host controller for the links connecting the endpoint. Software must also initialize the split transaction state bit (*SplitXState*) to **Do-Start-Split**. Finally, if the endpoint is a control endpoint, then system software must set the *Control Transfer Type* (*C*) bit in the queue head to a one. If this is not a control transfer type endpoint, the *C* bit must be initialized by software to be a zero. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the *C* bit is a zero, the split transaction token's ET field is set to indicate a bulk endpoint. When the *C* bit is a one, the split transaction token's ET field is set to indicate a control endpoint. Refer to Chapter 8 of USB Specification Revision 2.0 for details.

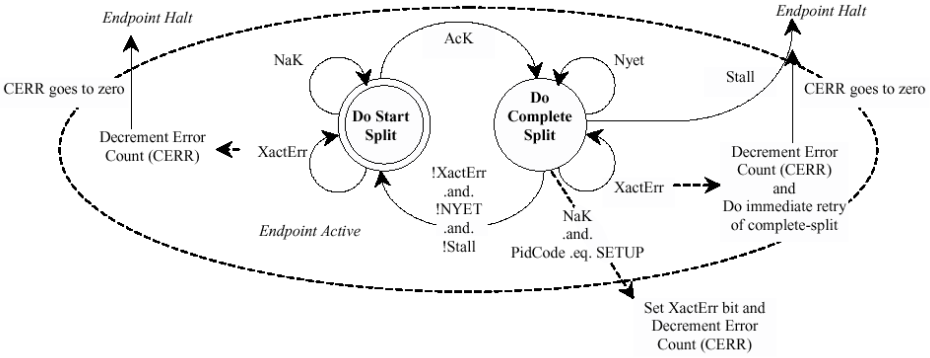


Figure 60-80. Host Controller Asynchronous Schedule Split-Transaction State Machine

Asynchronous - Do Start Split

This is the state which software must initialize a full- or low-speed asynchronous queue head. This state is entered from the **Do Complete Split** state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For queue heads in this state, the host controller will execute a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and *PidCode* indicates an IN or OUT transaction, then the host controller will reload the error counter (*CErr*). If it is a successful bus

transaction and the *PidCode* indicates a SETUP, the host controller will not reload the error counter. If the transaction translator responds with a Nak, the queue head is left in this state, and the host controller proceeds to the next queue head in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements *Cerr* and proceeds to the next queue head in the asynchronous schedule.

Asynchronous—Do Complete Split

This state is entered from the **Do Start Split** state only after a start-split transaction receives an Ack handshake from the transaction translator.

For queue heads in this state, the host controller will execute a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the queue head is left in this state, the error counter is reset and the host controller proceeds to the next queue head in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the queue head's *PidCode* indicates an IN or OUT, the host controller will reload the error counter (*CErr*). When a Nyet handshake is received for a complete-split bus transaction where the queue head's *PidCode* indicates a SETUP, the host controller must not adjust the value of *CErr.*

Independent of *PIDCode*, the following responses have the effects:

- Transaction Error (XactErr). Timeout or data CRC failure, etc. The error counter (*Cerr*) is decremented by one and the complete split transaction is *immediately* retried (if possible). If there is not enough time in the micro-frame to execute the retry, the host controller MUST ensure that the next time the host controller begins executing from the Asynchronous schedule, it must begin executing from this queue head. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A method to accomplish this behavior is to not advance the asynchronous schedule. When the host controller returns to the asynchronous schedule in the next micro-frame, the first transaction from the schedule will be the retry for this endpoint. If *Cerr* went to zero, the host controller must halt the queue.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced and the state is exited. If the *PidCode* is a SETUP, then the Nak response is a protocol error. The *XactErr* status bit is set to a one and the *CErr* field is decremented.
- STALL. The target endpoint responded with a STALL handshake. The host controller sets the *halt* bit in the status byte, retires the qTD but does not attempt to advance the queue.
- If the *PidCode* indicates an IN, then any of following responses are expected:
- DATA0/1. On reception of data, the host controller ensures the PID matches the expected data toggle and checks CRC. If the packet is *good*, the host controller will advance the state of the transfer, e.g. move the data pointer by the number of bytes received, decrement *BytesToTransfer* field by the number of bytes received, and toggle the *dt* bit. The host controller will then exit this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

- If the data sequence PID does not match the expected, the data is ignored, the transfer state is not advanced and this state is exited. If the *PidCode* indicates an OUT/SETUP, then any of following responses are expected:
- ACK. The target endpoint accepted the data, so the host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount and the data toggle bit (*dt*) is toggled. The host controller will then exit this state.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers via Queue Heads).

60.4.4.12.2 Split Transaction Interrupt

Split-transaction Interrupt-IN/OUT endpoints are managed via the same data structures used for high-speed interrupt endpoints. They both co-exist in the periodic schedule. Queue heads/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a high-speed endpoint, the host controller will visit a queue head, execute a high-speed transaction (if criteria are met) and advance the transfer state (or not) depending on the results of the entire transaction. For low- and full-speed endpoints, the details of the *execution* phase are different (i.e. takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, etc. occurs as defined in Section Managing Control/Bulk/Interrupt Transfers via Queue Heads, but only occurs on the completion of a split transaction.

Split Transaction Scheduling Mechanisms for Interrupt

Full- and low-speed Interrupt queue heads have an *EPS* field indicating full- or low-speed and have a non-zero *S-mask* field. The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. Low- and full-speed interrupt queue heads require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. Software should not set these fields to indicate the queue head is an interrupt unless the queue head is used in the periodic schedule.

System software manages the per/transaction translator periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each endpoint will occur. The characteristics of the transaction translator are such that the high-speed transaction protocol must execute during explicit micro-frames, or the data or response information in the pipeline is lost. Figure 60-81 illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule

and queue head data structure. The **S** and ^{C}x labels indicate micro-frames where software can schedule start-splits and complete splits (respectively).

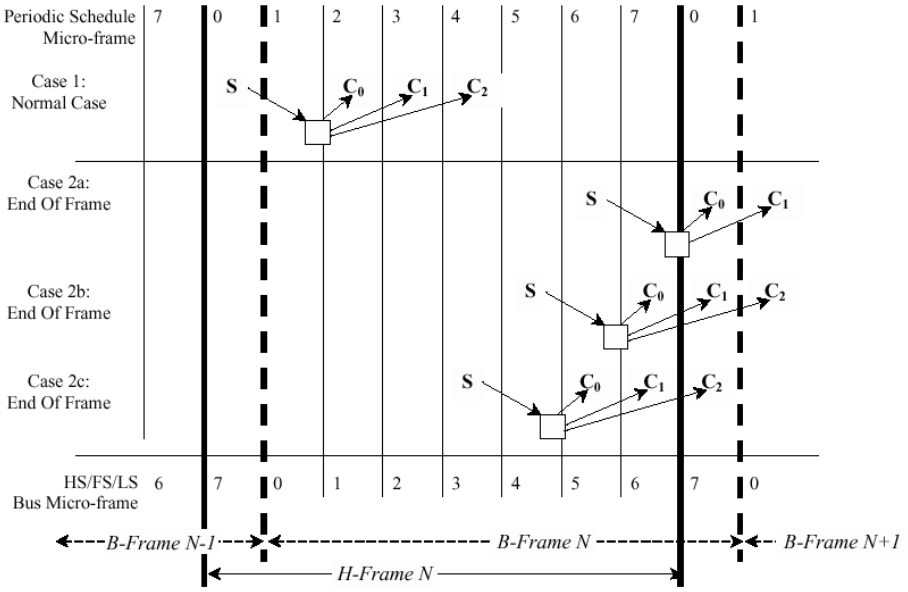


Figure 60-81. Split Transaction, Interrupt Scheduling Boundary Conditions

The scheduling cases are as follows:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (*H-Frame* in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, when and how many complete-splits must be scheduled to account for earliest to latest execution on the full/low-speed link. The complete-splits may span the *H-Frame* boundary when the start-split is in micro-frame 4 or later. When this occurs, the *H-Frame* to *B-Frame* alignment requires that the queue head be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs. Figure 60-82 illustrates the general layout of the periodic schedule.

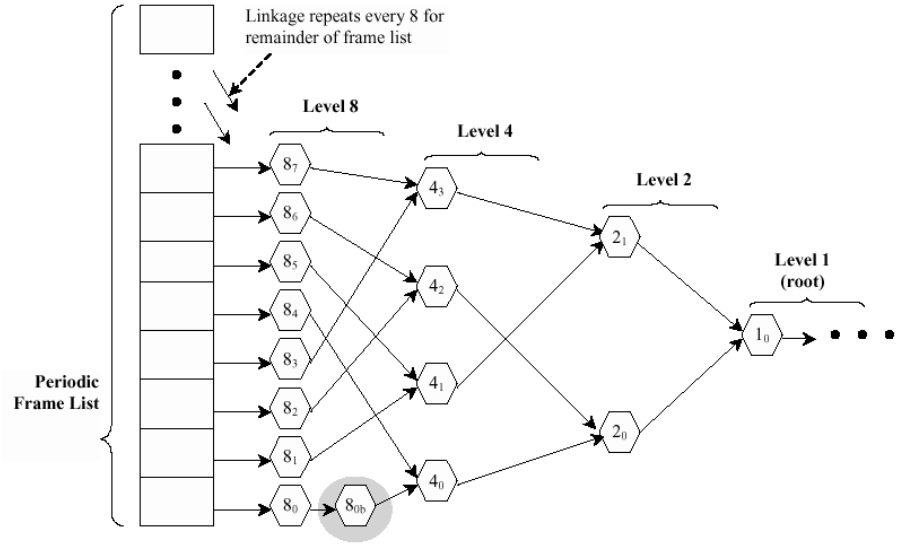


Figure 60-82. General Structure of EHCI Periodic Schedule Utilizing Interrupt Spreading

The periodic frame list is effectively the leaf level a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a 2^N poll rate. Software can efficiently manage periodic bandwidth on the USB by *spreading* interrupt queue heads that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate 8 queue heads and account for them once in the high-speed bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the queue head for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8_{0b} where such an endpoint. Without additional support on the interface, to get 8_{0b} reachable at the correct time, software would have to link 8_1 to 8_{0b}. It would then have to move 4_1 and everything linked after into the same path as 4_0. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. Section Host Controller Operational Model for FSTNs defines the hardware and software operational model requirements for using FSTNs.

The following queue head fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of a queue head (see Table 60-79). This bit is used to track the current state of the split transaction.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 60-81, case one, the *S-mask* would have a value of 00000001b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates **Do_Start**, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.

- *Frame C-mask*. This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to Figure 60-81, case one, the *C-mask* would have a value of 00011100b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates **Do_Complete**, and the current micro-frame as indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. It is software's responsibility to ensure that the translation between *H-Frames* and *B-Frames* is correctly performed when setting bits in *S-mask* and *C-mask*

Host Controller Operational Model for FSTNs

The FSTN data structure is used to manage Low/Full-speed interrupt queue heads that need to be reached from consecutive frame list locations (i.e. boundary cases 2a through 2c). An FSTN is essentially a *back pointer*, similar in intent to the back pointer field in the siTD data structure (see Section siTD Back Link Pointer).

This feature provides software a simple primitive to save a schedule position, redirect the host controller to traverse the necessary queue heads in the previous frame, then restore the original schedule position and complete normal traversal.

There are four components to the use of FSTNs:

- FSTN data structure.
- A *Save Place* indicator. This is always an FSTN with its *Back Path Link Pointer. T-bit* set to zero.
- A *Restore* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to a one.
- Host controller FSTN traversal rules.

Host Controller Operational Model for FSTNs

When the host controller encounters an FSTN during micro-frames 2 through 7 it simply follows the node's *Normal Path Link Pointer* to access the next schedule data structure. Note that the FSTN's *Normal Path Link Pointer.T-bit* may set to a one, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a *Save-Place* FSTN in micro-frames 0 or 1, it will save the value of the *Normal Path Link Pointer* and set an internal flag indicating that it is executing in *Recovery Path* mode. *Recovery Path* mode modifies the host controller's rules for how it traverses the schedule and limits which data structures will be considered for execution of bus transactions. The host controller continues executing in *Recovery Path* mode until it encounters a *Restore* FSTN or it determines that it has reached the end of the micro-frame (see details in the list below).

The rules for schedule traversal and limited execution while in *Recovery Path* mode are:

- Always follow the *Normal Path Link Pointer* when it encounters an FSTN that is a *Save-Place* indicator. The host controller must not recursively follow *Save-Place* FSTNs. Therefore, while executing in *Recovery Path* mode, it must never follow an FSTN's *Back Path Link Pointer.*
- Do not process an siTD or, iTD data structure. Simply follow its *Next Link Pointer.*

- Do not process a QH (Queue Head) whose *EPS* field indicates a high-speed device. Simply follow its *Horizontal Link Pointer.*
- When a QH's *EPS* field indicates a Full/Low-speed device, the host controller will only consider it for execution if its *SplitXState* is **DoComplete** (note: this applies whether the *PID Code* indicates an IN or an OUT). See Sections Execute Transaction and Tracking Split Transaction Progress for Interrupt Transfers for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction. Note that the host controller must not execute a Start-split transaction while executing in *Recovery Path* mode. See Section Periodic Isochronous—Do Complete Split for special handling when in *Recovery Path* mode.
- Stop traversing the *recovery path* when it encounters an FSTN that is a *Restore* indicator. The host controller unconditionally uses the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* when returning to the normal path traversal. The host controller must clear the context of executing a *Recovery Path* when it restores schedule traversal to the *Save-Place* FSTN's *Normal Path Link Pointer.*
- If the host controller determines that there is not enough time left in the micro-frame to complete processing of the periodic schedule, it abandons traversal of the recovery path, and clears the context of executing a recovery path. The result is that at the start of the next consecutive micro-frame, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in Figure 60-83.

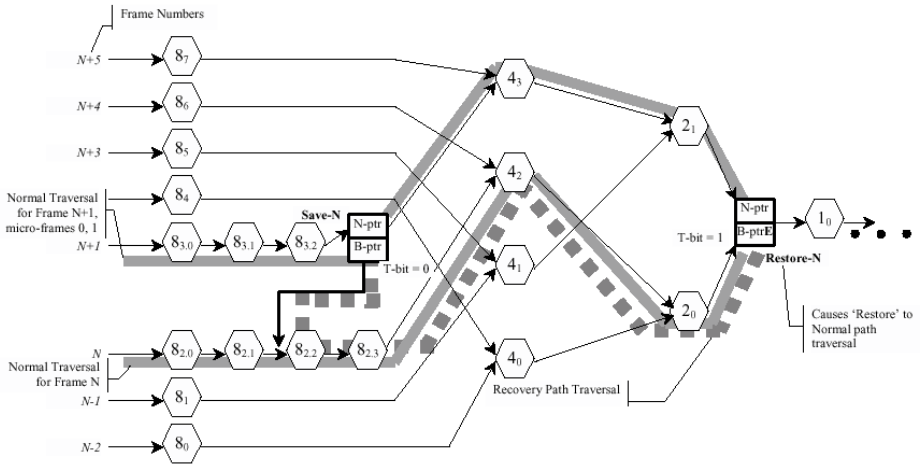


Figure 60-83. Example Host Controller Traversal of Recovery Path via FSTNs

In frame N+1 (micro-frames 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that Save-N.Back Path Link Pointer.T-bit is zero (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in *Recovery Path* mode (the recovery path is annotated in Figure 60-83 with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to a one (definition of a Restore indicator), so the host controller exits *Recovery Path* mode by clearing the internal *Recovery Path* mode flag and commences (restores) schedule traversal using the

saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* (e.g. Save-N.Normal Path Link Pointer). The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, **$8_{2.2}$**, **$8_{2.3}$**, **4_2**, **2_0**, **Restore-N**, 4_3, 2_1, Restore-N, 1_0 …}. The nodes on the recovery-path are bolded.In frame N (micro-frames 0-7), for this example, the host controller will traverse all of the schedule data structures utilizing the *Normal Path Link Pointers* in any FSTNs it encounters. This is because the host controller has not yet encountered a *Save-Place* FSTN so it not executing in *Recovery Path* mode. When it encounters the *Restore* FSTN, (Restore-N), during micro-frames 0 and 1, it uses Restore-N.Normal Path Link Pointer to traverse to the next data structure (i.e. normal schedule traversal). This is because the host controller must use a Restore FSTN's *Normal Path Link Pointer* when not executing in a *Recovery-Path* mode. The nodes traversed during frame N include: {$8_{2.0}$, $8_{2.1}$, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 1_0 …}.

In frame N+1 (micro-frames 2-7), when the host controller encounters Save-Path FSTN Save-N, it will unconditionally follow Save-N.Normal Path Link Pointer. The nodes traversed during these micro-frames include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, 4_3, 2_1, Restore-N, 1_0 …}.

Software Operational Model for FSTNs

Software must create a consistent, coherent schedule for the host controller to traverse. When using FSTNs, system software must adhere to the following rules:

- Each *Save-Place* indicator requires a matching *Restore* indicator.
  - The *Save-Place* indicator is an FSTN with a valid *Back Path Link Pointer and T-bit* equal to zero. Note that *Back Path Link Pointer.Typ* field must be set to indicate the referenced data structure is a queue head. The *Restore* indicator is an FSTN with its *Back Path Link Pointer.T-bit* set to a one.
  - A *Restore* FSTN may be matched to one or more *Save-Place* FSTNs. For example, if the schedule includes a poll-rate 1 level, then system software only needs to place a *Restore* FSTN at the beginning of this list in order to match all possible *Save-Place* FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more *Save-Place* FSTNs are used, then System Software must ensure the *Restore* FSTN's *Normal Path Link Pointer*'s *T-bit* is set to a one, as this will be use to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a *Restore* FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that *Recovery Path* mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A *Save-Place* FSTN's *Back Path Link Pointer* must reference a queue head data structure. The referenced queue head must be reachable from the previous frame list location. In other words, if the *Save-Place* FSTN is reachable from frame list offset N, then the FSTN's *Back Path Link Pointer* must reference a queue head that is reachable from frame list offset N-1.

Software should make the schedule as efficient as possible. What this means in this context is that software should have no more than one *Save-Place* FSTN reachable in any single frame. Note there will be times when two (or more, depending on the implementation) could exist as full/low-speed footprints change with bandwidth adjustments. This could occur, for example when a bandwidth rebalance causes system software to move the *Save-Place* FSTN from one poll rate level to another. During the transition, software must preserve the integrity of the previous schedule until the new schedule is in place.

Tracking Split Transaction Progress for Interrupt Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost. For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to get it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline. When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted and successfully executed by the USB 2.0 Hub, but the complete-splits get unrecoverable errors on the high-speed link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the right time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot get timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition, but rather effects only the progress of the transfer. The queue head has the following fields to track the progress of each split transaction. These fields are used to keep incremental state about which (and when) portions have been executed.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets one of the *C-prog*-mask bits for each complete-split executed. The bit position is determined by the micro-frame number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed then it means one (or more) have been skipped and data has potentially been lost.
- *FrameTag*. This field is used by the host controller during the complete-split portion of the split transaction to tag the queue head with the frame number (*H-Frame* number) when the next complete split must be executed.
- *S-bytes*. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The *S-bytes* field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

Split Transaction Execution State Machine for Interrupt

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame*.

As with asynchronous Full- and Low-speed endpoints, a split-transaction state machine is used to manage the split transaction sequence. Aside from the fields defined in the queue head for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- *cMicroFrameBit*. This is a single-bit encoding of the current micro-frame number. It is an eight-bit value calculated by the host controller at the beginning of every micro-frame. It is calculated from the three least significant bits of the *FRINDEX* register (i.e. *cMicroFrameBit* = (1

shifted-left(*FRINDEX*[2:0]))). The *cMicroFrameBit* has at most one bit asserted, which always corresponds to the current micro-frame number. For example, if the current micro-frame is 0, then *cMicroFrameBit* will equal 00000001b. The variable *cMicroFrameBit* is used to compare against the *S-mask* and *C-mask* fields to determine whether the queue head is marked for a start- or complete-split transaction for the current micro-frame.

Figure 60-84 illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the *SplitXState* is at **Do_Start** and the single bit in *cMicroFrameBit* has a corresponding bit active in *QH.S-mask*. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to **Do_Complete**. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the **Do_Complete** state. This translates simply to the fact that there are multiple bits set to a one in the *QH.C-mask* field.

The host controller keeps the queue head in the **Do_Complete** state until the split transaction is complete (see definition below), or an error condition triggers the *three-strikes-rule* (e.g. after the host tries the same transaction three times, and each encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).

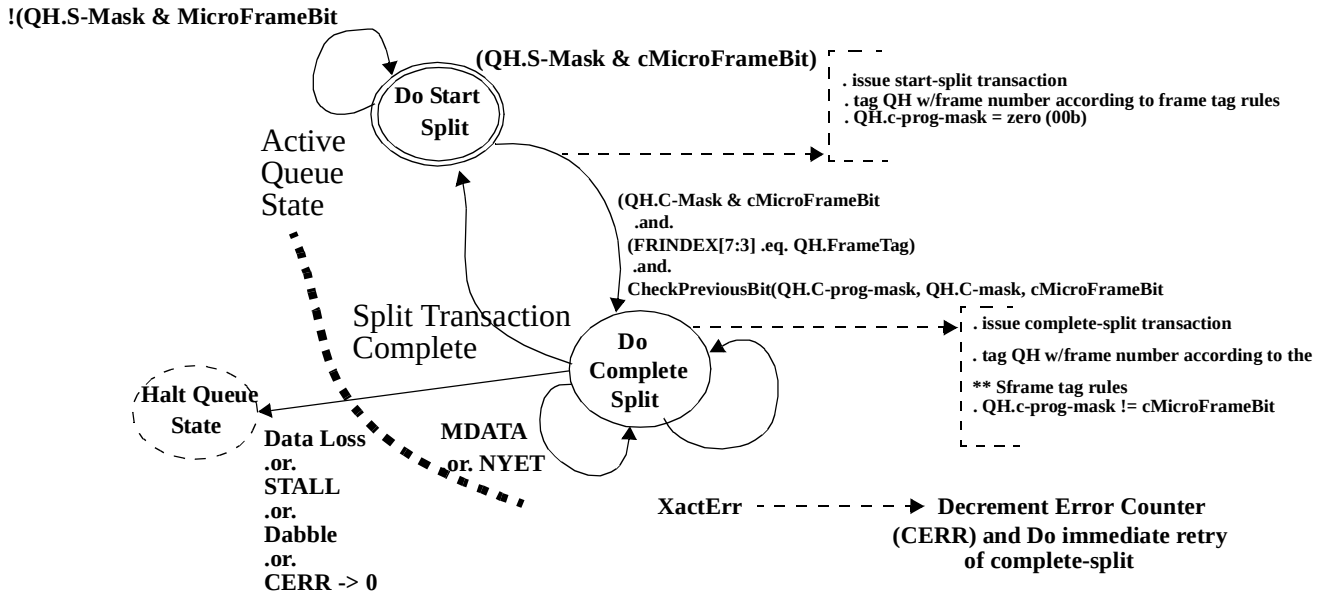


Figure 60-84. Split Transaction State Machine for Interrupt

See Previous Section for the frame tag management rules.

Periodic Interrupt—Do Start Split

This is the state software must initialize a full- or low-speed interrupt queue head *StartXState* bit. This state is entered from the **Do_Complete Split** state only after the split transaction is complete. This occurs when

one of the following events occur: The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the full- or low-speed endpoint's NAK response.
- ACK. An ACK response is a propagation of the full- or low-speed endpoint's ACK response. Only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the low-/full-speed link below the transaction translator had a failure (e.g. timeout, bad CRC, etc.).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the full- or low-speed endpoint, see Section Periodic Isochronous—Do Complete Split for the definition of ‘Last’.

Each time the host controller visits a queue head in this state (once within the **Execute Transaction** state), it performs the following test to determine whether to execute a start-split.

- *QH.S-mask* is bit-wise anded with *cMicroFrameBit*.

If the result is non-zero, then the host controller will issue a start-split transaction. If the *PIDCode* field indicates an IN transaction, the host controller must zero-out the *QH.S-bytes* field. After the split-transaction has been executed, the host controller sets up state in the queue head to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into *QH.FrameTag* field (see Section), set *C-prog-mask* to zero (00h), and exits this state. Note that the host controller must not adjust the value of *CErr* as a result of completion of a start-split transaction.

Periodic Interrupt - Do Complete Split

This state is entered unconditionally from the **Do Start Split** state after a start-split transaction is executed on the bus. Each time the host controller visits a queue head in this state (once within the **Execute Transaction** state), it checks to determine whether a complete-split transaction should be executed now.

There are four tests to determine whether a complete-split transaction should be executed.

- **Test A**. *cMicroFrameBit* is bit-wise anded with *QH.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame.
- **Test B**. *QH.FrameTag* is compared with the current contents of *FRINDEX*[7:3]. An equal indicates a match.
- **Test C**. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)

Begin

-- Return values:

-- TRUE - no error

-- FALSE - error
```

```
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous micro-frame. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask)then
                          If not(previousBit bitAND QH.C-prog-mask) then
                 rvalue = FALSE;
         End if
End If
-- If the C-prog-mask already has a one in this bit position,
-- then an aliasing
-- error has occurred. It will probably get caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- should not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
 rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- **Test D.** Check to see if a start-split should be executed in this micro-frame. Note this is the same test performed in the **Do Start Split** state (see Section Periodic Isochronous - Do Start Split). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a *Recovery Path* mode, it means a start-split should occur in this micro-frame. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If (A .and. B .and. C .and. not(D)) then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. On completion of the complete-split transaction, the host controller

records the result of the transaction in the queue head and sets *QH.FrameTag* to the expected *H-Frame* number (see Section). The effect to the state of the queue head and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the effects (note that any responses that result in decrementing of the *CErr* will result in the queue head being halted by the host controller if the result of the decrement is zero):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition where all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the queue head is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the clompete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the **Last** complete split can be performed by XOR *QH.C-mask* with *QH.C-prog-mask*. If the result is all zeros then all complete-splits have been executed. When this condition occurs, the *XactErr* status bit is set to a one and the *CErr* field is decremented.
- NYET (and not Last). See above description for testing for **Last**. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask* and *FrameTag*) and stay in this state. The host controller must not adjust *CErr* on this response.
- Transaction Error (XactErr). Timeout, data CRC failure, etc. The *CErr* field is decremented and the *XactErr* bit in the *Status* field is set to a one. The complete split transaction is *immediately* retried (if *Cerr* is non-zero).If there is not enough time in the micro-frame to complete the retry and the endpoint is an IN, or *CErr* is decremented to a zero from a one, the queue is halted. If there is not enough time in the micro-frame to complete the retry and the endpoint is an OUT and *CErr* is not zero, then this state is exited (i.e. return to **Do Start Split**). This results in a retry of the entire OUT split transaction, at the next poll period. Refer to Chapter 11 Hubs (specifically the section full- and low-speed Interrupts) in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint ACK'd the data and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount. And the data toggle bit (*dt*) is toggled. The host controller will then exit this state for this queue head. The host controller must reload *CErr* with maximum value on this response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers via Queue Heads).
- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in *QH.S-bytes*. The host controller must not adjust *CErr* on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in *QH.S-bytes*. The state of the transfer is advanced by the result and the host controller will exit this state for this queue head.

- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section Managing Control/Bulk/Interrupt Transfers via Queue Heads).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced and this state is exited.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload *CErr* with maximum value on this response.
- ERR. There was an error during the full- or low-speed transaction. The ERR status bit is set to a one, *Cerr* is decremented, the state of the transfer is not advanced, and this state is exited.
- STALL. The queue is halted (an exit condition of the **Execute Transaction** state). The status field bits: *Active* bit is set to zero and the *Halted* bit is set to a one and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that data or response was lost. Table 60-99 lists the possible combinations and the appropriate action.

Table 60-99. Interrupt IN/OUT Do Complete Split State Execution Criteria

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current micro-frame.Host controller should continue walking the schedule.
A not(C)	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	Progress bit check failed. These means a complete-split has been missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (e.g. start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A not(B) C	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	*QH.FrameTag* test failed. This means that exactly one or more *H-Frames* have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (e.g. start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one.
A B C not(D)	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.
D	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to fire. If *PIDCode* is an IN, then the Queue head must be halted. If *PIDCode* is an OUT, then the transfer state is not advanced and the state exited (e.g. start-split is retried). This is a host-induced error and does not effect *CERR*. In either case, set the *Missed Micro-frame* bit in the status field to a one. Note: When executing in the context of a *Recovery Path* mode, the host controller is allowed to process the queue head and take the actions indicated above, or it may wait until the queue head is visited in the normal processing mode. Regardless, the host controller must not execute a start-split in the context of a executing in a *Recovery Path* mode.

Managing QH.FrameTag Field

The *QH.FrameTag* field in a queue head is completely managed by the host controller. The rules for setting *QH.FrameTag* are simple:

- **Rule 1:** If transitioning from **Do Start Split** to **Do Complete Split** and the current value of *FRINDEX*[2:0] is 6 *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different *H-Frames* (case 2a, see Figure 60-81).
- **Rule 2:** If the current value of *FRINDEX*[2:0] is 7, *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates staying in **Do Complete Split** for cases 2a, 2b, and 2c (Figure 60-81).
- **Rule 3:** If transitioning from **Do_Start Split** to **Do Complete Split** and the current value of *FRINDEX*[2:0] is not 6, or currently in **Do Complete Split** and the current value of (*FRINDEX*[2:0]) is not 7, *FrameTag* is set to *FRINDEX*[7:3]. This accommodates all other cases (Figure 60-81).

Rebalancing the Periodic Schedule

System software must occasionally adjust a periodic queue head's S-mask and C-mask fields during operation. This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more queue head's are assigned new execution footprints (i.e. new S-mask and C-mask values).

It is imperative that System software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI host controller provides a simple assist to system software. System software sets the *Inactivate-on-next-Transaction* (*I*) bit to a one to signal the host controller that it intends to update the S-mask and C-mask on this queue head. System software will then wait for the host controller to observe the *I-bit* is a one and transition the *Active* bit to a zero. The rules for how and when the host controller sets the *Active* bit to zero are enumerated below:

- If the *Active* bit is a zero, no action is taken. The host controller does not attempt to advance the queue when the *I-bit* is a one.
- If the *Active* bit is a one and the *SplitXState* is **DoStart** (regardless of the value of *S-mask*), the host controller will simply set *Active* bit to a zero. The host controller is not required to write the transfer state back to the *current* qTD. Note that if the *S-mask* indicates that a start-split is scheduled for the current micro-frame, the host controller must not issue the start-split bus transaction. It must set the *Active* bit to zero.

System software must save transfer state before setting the *I-bit* to a one. This is required so that it can correctly determine what transfer progress (if any) occurred after the *I-bit* was set to a one and the host controller executed it's final bus-transaction and set *Active* to a zero.

After system software has updated the S-mask and C-mask, it must then reactivate the queue head. Since the *Active* bit and the *I-bit* cannot be updated with the same write, system software needs to use the following algorithm to coherently re-activate a queue head that has been stopped via the *I-bit*.

1. Set the *Halted* bit to a one.
2. Set the *I-bit* to a zero.
3. Set the *Active* bit to a one and the *Halted* bit to a zero in the same write.

Setting the *Halted* bit to a one inhibits the host controller from attempting to advance the queue between the time the *I-bit* goes to a zero and the *Active* bit goes to a one.

60.4.4.12.3 Split Transaction Isochronous

Full-speed isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller utilizes siTD data structure to support the special requirements of isochronous split-transactions. This data structure uses the scheduling model of isochronous TDs (iTD, Section Isochronous (High-Speed) Transfer Descriptor (iTD)) (see Section Managing Isochronous Transfers Using iTDs for the operational model of iTDs) with the contiguous data feature provided by queue heads. This simple arrangement allows a single isochronous scheduling model and adds the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.

Split Transaction Scheduling Mechanisms for Isochronous

Full-speed isochronous transactions are managed through a transaction translator's periodic pipeline. As with full- and low-speed interrupt, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each full-speed isochronous endpoint occur. The requirements described in Section Split Transaction Scheduling Mechanisms for Interrupt apply. Figure 60-85 illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The ^{s}x and ^{c}x labels indicate micro-frames where software can schedule start- and complete-splits (respectively). The *H-Frame* boundaries are marked with a large, solid bold vertical line. The *B-Frame* boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the *H-Frame*.

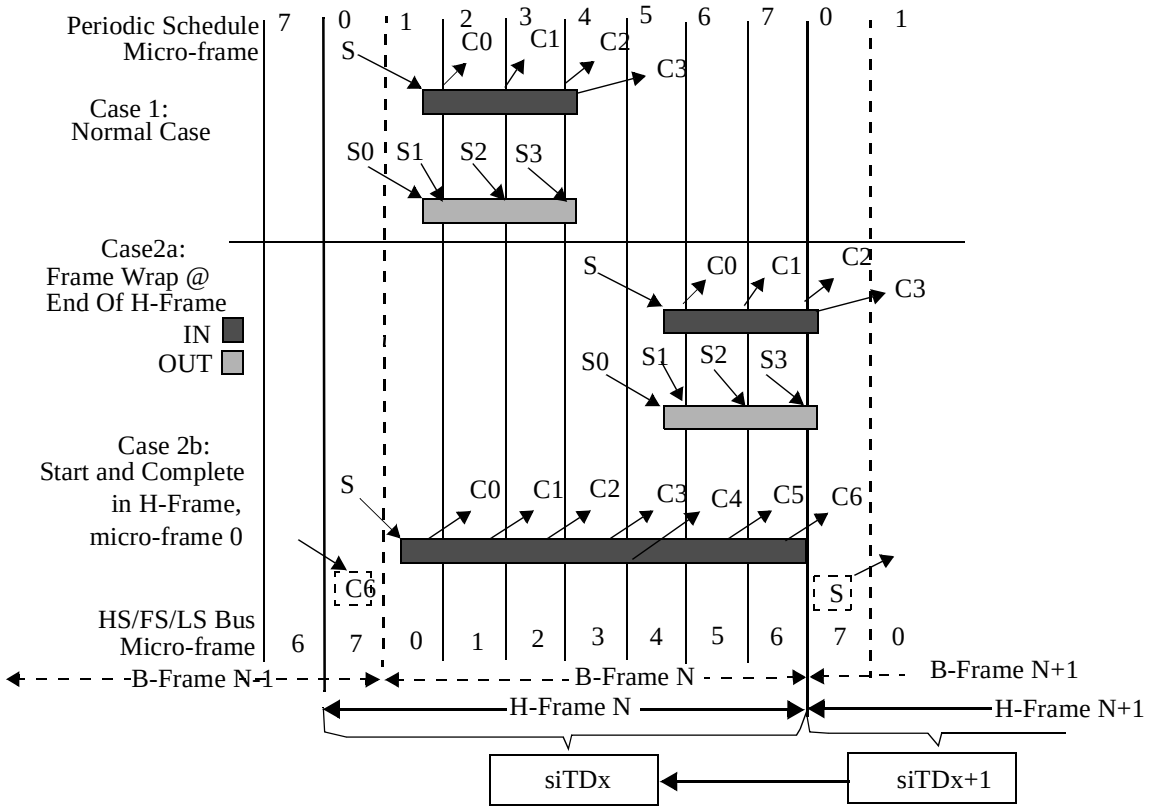


Figure 60-85. Split Transaction, Isochronous Scheduling Boundary Conditions

When the endpoint is an isochronous OUT, there are only start-splits, and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to *N* complete-splits. The scheduling boundary cases are:

- *Case 1*: The entire split transaction is completely bounded by an *H-Frame*. For example: the start-splits and complete-splits are all scheduled to occur in the same *H-Frame*.
- *Case 2a*: This boundary case is where one or more (at most two) complete-splits of a split transaction IN are scheduled across an *H-Frame* boundary. This can only occur when the split transaction has the possibility of moving data in *B-Frame*, micro-frames 6 or 7 (*H-Frame* micro-frame 7 or 0). When an *H-Frame* boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list.(e.g. it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction).
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each full-speed IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- Software must never schedule full-speed isochronous OUTs across an *H-Frame* boundary.
- *Case 2b*: This case can only occur for a very large isochronous IN. It is the only allowed scenario where a start-split and complete-split for the same endpoint can occur in the same micro-frame. Software must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579 byte maximum packet size.

A subset of the same mechanisms employed by full- and low-speed interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by system software to instruct the host controller when to execute portions of the split transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of an siTD (see Table 60-74). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in Section Split Transaction Execution State Machine for Interrupt .
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 60-85, case one, the *S-mask* would have a value of 00000001b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates **Do Start Split**, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask*. This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in Figure 60-85, case one, the *C-mask* would have a value of 00111100b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates **Do Complete Split**, and the current micro-frame as indicated by *FRINDEX*[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.
- *Back Pointer*. This field in a siTD is used to complete an IN split-transaction using the previous *H-Frame*'s siTD. This is only used when the scheduling of the complete-splits span an *H-Frame* boundary.

There exists a one-to-one relationship between a high-speed isochronous split transaction (including all start- and complete-splits) and one full-speed isochronous transaction. An siTD contains (amongst other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one full-speed isochronous data payload. This means that for any full-speed transaction payload, a single siTD's data buffer must be used. This rule applies to both IN an OUTs. An siTD's scheduling information usually also maps to one high-speed isochronous split transaction. The exception to this rule is the *H-Frame* boundary wrap cases mentioned above.

The siTD data structure describes at most, one frame's worth of high-speed transactions and that description is strictly bounded within a frame boundary. Figure 60-86 illustrates some examples. On the top are examples of the full-speed transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the *B-Frames* (HS/FS/LS Bus) and the *H-Frames*. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each *H-Frame* corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.

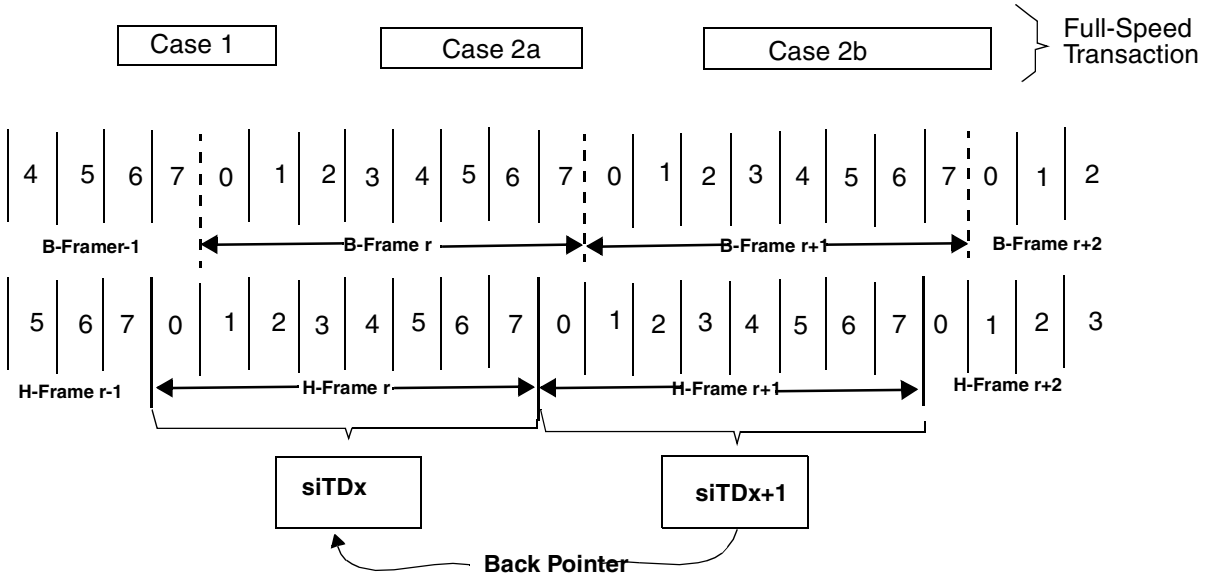


Figure 60-86. siTD Scheduling Boundary Examples

Each case is described below:

- *Case 1*: One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single *H-Frame*.
- *Case 2a, 2b*: Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete the scheduling of the isochronous split transaction. $siTD_X$ is used to always issue the start-split and the first *N* complete-splits. The full-speed transaction (for these cases) can deliver data on the full-speed bus segment during micro-frame 7 of *H-Frame*$_{Y+1}$, or micro-frame 0 of *H-Frame*$_{Y+2}$. The complete splits are scheduled using $siTD_{X+2}$ (not shown). The complete-splits to extract this data must use the buffer pointer from $siTD_{X+1}$. The only way for the host controller to reach $siTD_{X+1}$ from *H-Frame*$_{Y+2}$ is to use $siTD_{X+2}$'s back pointer. The host controller rules for when to use the back pointer are described is Section Periodic Isochronous—Do Complete Split.

Software must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- Software must ensure that an isochronous split-transaction is started so that it will complete before the end of the *B-Frame*.
- Software must ensure that for a single full-speed isochronous endpoint, there is never a start-split and complete-split in *H-Frame, micro-frame 1*. This is mandated as a rule so that case 2a and case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split was in micro-frame 1 of *H-Frame* N and the last complete-split would need to occur in micro-frame 1 of H-Frame N+1. However, it is impossible to discriminate between cases 2a and case 2b, which has significant impact on the complexity of the host controller.

Tracking Split Transaction Progress for Isochronous Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device to host data is lost. Isochronous endpoints do not employ the concept of a halt on error, however the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped micro-frames), timeouts and corrupted data received.

In similar kind to interrupt split-transactions, the portions of the split transaction protocol must execute in the micro-frames they are scheduled. The queue head data structure used to manage full- and low-speed interrupt has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs, for their transfers, and the data structures are only reachable via the schedule in the exact micro-frame in which they are required (so all the mechanism employed for tracking in queue heads is not required for siTDs). Software has the option of reusing siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction N are consumed and the siTD reinitialized (activated) before the host controller gets back to the siTD (in a future micro-frame).

Split-transaction isochronous OUTs utilize a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields *Transaction Position* (*TP*) and *Transaction Count* (*T-count*). If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and handled by the transaction translator. See Section Periodic Isochronous - Do Start Split for a description on how these fields are used during a sequence of start-split transactions.

The fields *siTD.T-Count* and *siTD.TP* are used by the host controller to drive and sequence the transaction position annotations. It is the responsibility of system software to properly initialize these fields in each siTD. Once the budget for a split-transaction isochronous endpoint is established, *S-mask*, *T-Count*, and *TP* initialization values for all the siTD associated with the endpoint are constant. They remain constant until the budget for the endpoint is recalculated by software and the periodic schedule adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with split transaction Interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets a bit for each complete-split executed. The bit position is determined by the micro-frame (FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means one (or more) have been skipped and data has potentially been lost. System software is required to initialize this field to zero before setting an siTD's *Active* bit to a one.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. Refer to Section Asynchronous—Do Complete Split for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field *Total Bytes to Transfer* to decrement to zero. This response can occur, before all of the scheduled complete-splits have been executed. In other interface, data structures (e.g. high-speed data streams through queue heads), the transition of *Total Bytes to Transfer* to zero signals the end of the transfer and results in setting of the *Active* bit to zero. However, in this case, the result has not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary the periodic pipeline rules require that on a micro-frame boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the full-speed bus pipe stage and give the remaining bytes to the high-speed pipeline stage. At the micro-frame boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next micro-frame, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the full-speed pipeline stage). This could cause the siTD to decrement it's *Total Bytes to Transfer* field to zero, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the full-speed transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the *C-prog-mask* is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (e.g. a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then system software must detect that the siTDs have not been processed by the host controller (e.g. state not advanced) and report the appropriate error to the client driver.

Split Transaction Execution State Machine for Isochronous

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame*.

If the *Active* bit in the *Status* byte is a zero, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Section Tracking Split Transaction Progress for Interrupt Transfers , plus the variable *cMicroFrameBit* defined in Section Split Transaction Execution State Machine for Interrupt to track the progress of an isochronous split transaction. Figure 60-87 illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the *Active* bit in the *Status* field of a siTD. The Bold, dotted arcs denote the transitions between these states.

Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.

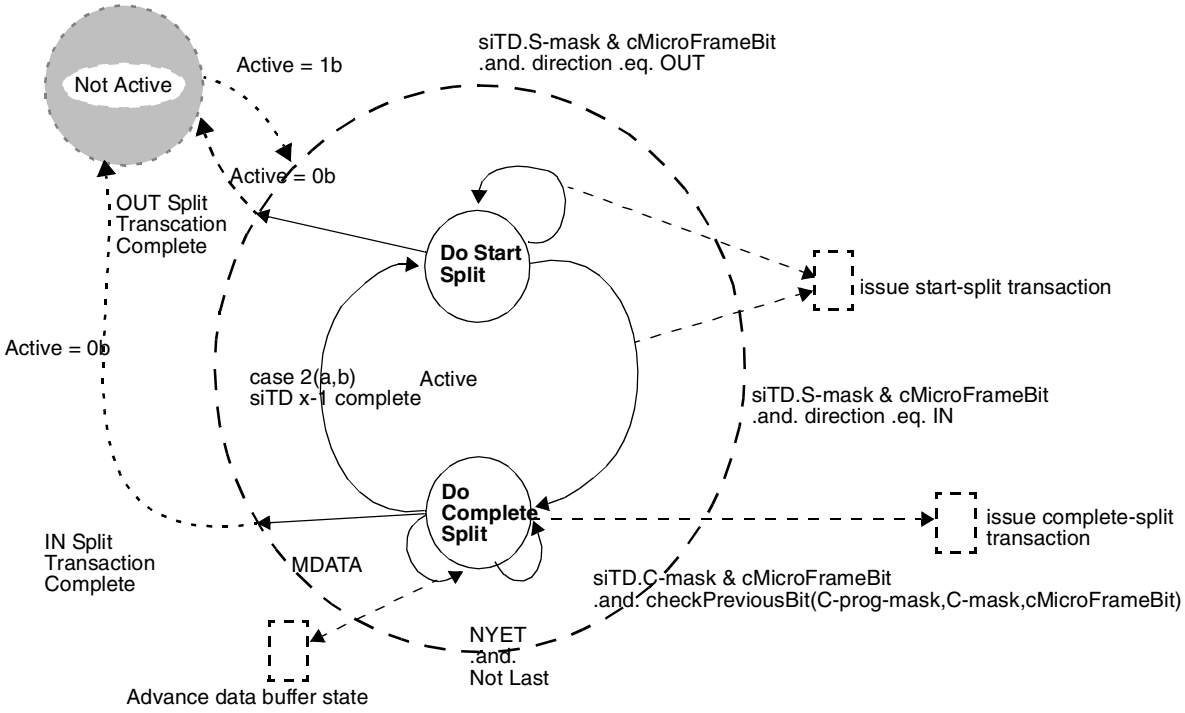


Figure 60-87. Split Transaction State Machine for Isochronous

Periodic Isochronous - Do Start Split

Isochronous split transaction OUTs use only this state. An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from **Do Complete Split** when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the *siTD.S-mask* against *cMicroFrameBit*. If there is a one in the appropriate position, the siTD will execute a start-split transaction. By definition, the host controller cannot *reach* an siTD at the wrong time. If the *I/O* field indicates an IN, then the start-split transaction includes only the extended token plus the full-speed token. Software must initialize the *siTD.Total Bytes To Transfer* field to the number of bytes expected. This is usually the maximum packet size for the full-speed endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (i.e. the *I/O* field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT it includes a data payload in the start-split transaction. The memory buffer address for the data payload is constructed by concatenating *siTD.Current Offset* with the page pointer indicated by the page selector field (*siTD.P*). A zero in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer

crosses a page boundary during the transaction, the host controller must detect the page cross, update the *siTD.P*-bit from a zero to a one, and begin using the *siTD.Page 1* with *siTD.Current Offset* as the memory address pointer. The field *siTD.TP* is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases the host controller simply uses the value in *siTD.TP* to mark the start-split with the correct transaction position code.

T-Count is always initialized to the number of start-splits for the current frame. *TP* is always initialized to the first required transaction position identifier. The scheduling boundary case (see Figure 60-86) is used to determine the initial value of *TP*. The initial cases are summarized in Table 60-100.

Table 60-100. Initial Conditions for OUT siTD's TP and T-count Fields

Case	T-count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates *T-Count* and *TP* appropriately so that the next start-split is correctly annotated. Table 60-101 illustrates all of the *TP* and *T-count* transitions, which must be accomplished by the host controller.

Table 60-101. Transaction Position (TP)/Transaction Count (T-Count) Transition Table

TP	T-count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when *T-count* starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when *T-count* starts at greater than 2.
MID	!=1	MID	*TP* stays at MID while *T-count* is not equal to 1 (e.g. greater than 1). This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the *T-count* starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state the following operations take place:

- The *siTD.Total Bytes To Transfer* and the *siTD.Current Offset* fields are adjusted to reflect the number of bytes transferred.
- The *siTD.P* (page selector) bit is updated appropriately.
- The *siTD.TP* and *siTD.T-count* fields are updated appropriately as defined in Table 60-101.

These fields are then written back to the memory based siTD. The *S-mask* is fixed for the life of the current budget. As mentioned above, *TP* and *T-count* are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of *S-mask*, the actual number of start-split transactions depends on *T-count* (or equivalently, *Total Bytes to Transfer*). The host controller must set the *Active* bit to

a zero when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when *T-Count* decrements to zero as a result of a start-split bus transaction. Equivalently, the host controller can detect when *Total Bytes to Transfer* decrements to zero. Either implementation must ensure that if the initial condition is *Total Bytes to Transfer* equal to zero and *T-count* is equal to a one, then the host controller will issue a single start-split, with a zero-length data payload. Software must ensure that *TP*, *T-count* and *Total Bytes to Transfer* are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (i.e. the transaction position annotation will be incorrect as received by the transaction translator).

Example scenarios are described in Section Split Transaction for Isochronous—Processing Examples.

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the *siTD.C-prog-mask* as it executes each scheduled start-split. The *checkPreviousBit*() algorithm defined in Section Periodic Isochronous—Do Complete Split can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed micro-frames. It can then set the siTD's *Active* bit to zero and stop execution of this siTD. This saves on both memory and high-speed bus bandwidth.

Periodic Isochronous—Do Complete Split

This state is only used by a split-transaction isochronous IN endpoint. This state is entered unconditionally from the **Do Start State** after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence they are applied depends on which micro-frame the host controller is currently executing which means that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- **Test A**. *cMicroFrameBit* is bit-wise anded with *siTD.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame. This test is always applied to a newly fetched siTD that is in this state.
- **Test B**. The *siTD.C-prog-mask* bit vector is checked to determine whether the previous complete splits have been executed. An example algorithm is below (this is slightly different than the algorithm used in Section Periodic Isochronous—Do Complete Split). The sequence in which this test is applied depends on the current value of FRINDEX[2:0]. If FRINDEX[2:0] is 0 or 1, it is not applied until the back pointer has been used. Otherwise it is applied immediately.

Algorithm Boolean CheckPreviousBit(`siTD.C-prog-mask, siTD.C-mask, cMicroFrameBi`**t)**

```
Begin

        Boolean rvalue = TRUE;

        previousBit = cMicroFrameBit rotate-right(1)

        -- Bit-wise anding previousBit with C-mask indicates whether there was an intent

        -- to send a complete split in the previous micro-frame. So, if the
```

```
        -- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
        -- happened.
        if previousBit bitAND siTD.C-mask then
                if not (previousBit bitAND siTD.C-prog-mask) then
                        rvalue = FALSE
                End if
        End if
        Return rvalue
End Algorithm
```

If Test A is true and FRINDEX[2:0] is zero or one, then this is a case 2a or 2b scheduling boundary (see Figure 60-85). See Section , Periodic Isochronous—Do Complete Split," for details in handling this condition.

If Test A and Test B evaluate to true, then the host controller will execute a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must do the following:

- Decrement the number of bytes received from *siTD.Total Bytes To Transfer,*
- Adjust *siTD.Current Offset* by the number of bytes received,
- Adjust *siTD.P* (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the *siTD.Status* field, depending on the results of the transaction.

Note that if the host controller encounters a condition where *siTD.Total Bytes To Transfer* is zero, and it receives more data, the host controller must not write the additional data to memory. The *siTD.Status.Active* bit must be set to zero and the *siTD.Status.Babble Detected* bit must be set to a one. The fields *siTD.Total Bytes To Transfer*, *siTD.Current Offset*, and *siTD.P* (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of *siTD.Total Bytes To Transfer*) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set *siTD.Status Active* to a zero and *siTD.Status.Babble Detected* to a one when it receives and MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have the noted effects:

- ERR. The full-speed transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the *ERR* bit in the *siTD.Status* field and sets the *Active* bit to a zero.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, etc. The *siTD.Status* field *XactErr* field is set to a one and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry

more than two times. If the host controller exhausts the retries or the end of the micro-frame occurs, the *Active* bit is set to zero.

- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the *Active* bit is set to a zero. If the *Bytes To Transfer* field has not decremented to zero (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for was actually received. This *short packet* event does not set the USBINT status bit in the USBSTS register to a one. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. Last was defined in Section Periodic Isochronous—Do Complete Split. If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the *Active* bit is set to a zero. No bits are set in the *Status* field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled clompete-splits with NYETs, meaning that the start-split issued by the host controller was not received. This result should be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.
- MDATA (and Last). See above description for testing for **Last**. This can only occur when there is an error condition. Either there has been a babble condition on the full-speed link, which delayed the completion of the full-speed transaction, or software set up the *S-mask* and/or *C-mask*s incorrectly. The host controller must set *XactErr* bit to a one and the *Active* bit is set to a zero.
- NYET (and not Last). See above description for testing for **Last**. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask*) and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the full-speed transaction data payload spans from micro-frame X to X+1 and during micro-frame X, the transaction translator will respond with an MDATA and the data accumulated up to the end of micro-frame X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the *Missed Micro-Frame* status bit and sets the *Active* bit to a zero.

Complete-Split for Scheduling Boundary Cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see Figure 60-85) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. Table 60-102 enumerates the transaction state fields.

Table 60-102. Summary siTD Split Transaction State

Buffer State	Status	Execution Progress
Total Bytes To Transfer P (page select) Current Offset TP (transaction position) T-count (transaction count)	All bits in the status field	C-prog-mask
Note: *TP* and *T-count* are used only for Host to Device (OUT) endpoints.		

If software has budgeted the schedule of this data stream with a frame wrap case, it must initialize the *siTD.Back Pointer* field to reference a valid siTD and have the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field set to a zero. Otherwise, software must set the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field to a one. The host controller's rules for interpreting when to use the *siTD.Back Pointer* field are listed below. These rules apply only when the siTD's *Active* bit is a one and the *SplitXState* is **Do Complete Split**.

- When cMicroFrameBit is a 1h and the siTDX.Back Pointer.T-bit is a zero
- If cMicroFrameBit is a 2h and siTDX.S-mask[0] is a zero

When either of these conditions apply, the host controller must use the transaction state from $siTD_{X-1}$.

To access $siTD_{X-1}$, the host controller reads on-chip the siTD referenced from $siTD_X$*.Back Pointer.*

The host controller must save the entire state from $siTD_X$ while processing $siTD_{X-1}$. This is to accommodate for case 2b processing. The host controller must not recursively walk the list of *siTD.Back Pointers.*

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is **Do Complete Split**), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see Table 60-102) of $siTD_{X-1}$ is appropriately advanced based on the results and written back to memory. If the resultant state of $siTD_{X-1}$'s *Active* bit is a one, then the host controller returns to the context of $siTD_X$, and follows its next pointer to the next schedule item. No updates to $siTD_X$ are necessary.

If $siTD_{X-1}$ is active (*Active* bit is a one and *SplitXStat* is **Do Start Split**), then the host controller must set *Active* bit to a zero and *Missed Micro-Frame* status bit to a one and the resultant status written back to memory.

If $siTD_{X-1}$'s *Active* bit is a zero, (because it was zero when the host controller first visited $siTD_{X-1}$ via $siTD_X$'s back pointer, it transitioned to zero as a result of a detected error, or the results of $siTD_{X-1}$'s complete-split transaction transitioned it to zero), then the host controller returns to the context of $siTD_X$ and transitions its *SplitXState* to **Do Start Split**. The host controller then determines whether the case 2b

start split boundary condition exists (i.e. if *cMicroframeBit* is a 1b and $siTD_X$.*S-mask*[0] is a 1b). If this criterion is met the host controller immediately executes a start-split transaction and appropriately advances the transaction state of $siTD_X$, then follows $siTD_X$.*Next Pointer* to the next schedule item. If the criterion is not met, the host controller simply follows $siTD_X$.*Next Pointer* to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of $siTD_{X-1}$ will have its *Active* bit set to zero when the host controller returns to the context of $siTD_X$. Also, note that software should not initialize an siTD with *C-mask* bits 0 and 1 set to a one and an *S-mask* with bit zero set to a one. This scheduling combination is not supported and the behavior of the host controller is undefined.

Split Transaction for Isochronous—Processing Examples

There is an important difference between how the hardware/software manages the isochronous split transaction state machine and how it manages the asynchronous and interrupt split transaction state machines. The asynchronous and interrupt split transaction state machines are encapsulated within a single queue head. The progress of the data stream depends on the progress of each split transaction. In some respects, the split-transaction state machine is sequenced via the **Execute Transaction** queue head traversal state machine (see Figure 60-78).

Isochronous is a pure time-oriented transaction/data stream. The interface data structures are optimized to efficiently describe transactions that need to occur at specific times. The isochronous split-transaction state machine must be managed across these time-oriented data structures. This means that system software must correctly describe the scheduling of split-transactions across more than one data structure.

Then the host controller must make the appropriate state transitions at the appropriate times, in the correct data structures.

For example, Table 60-103 illustrates a couple of frames worth of scheduling required to schedule a case 2a full-speed isochronous data stream.

Table 60-103. Example Case 2a - Software Scheduling siTDs for an IN Endpoint

siTDx		Micro-Frames								Initial SplitXState
#	Masks	0	1	2	3	4	5	6	7	
X	S-Mask					1				Do Start Split
	C-Mask	1	1					1	1	
X+1	S-Mask					1				Do Complete Split
	C-Mask	1	1					1	1	
X+2	S-Mask					1				Do Complete Split
	C-Mask	1	1					1	1	
X+3	S-Mask	Repeats previous pattern								Do Complete Split
	C-Mask									

This example shows the first three siTDs for the transaction stream. Since this is the case-2a frame-wrap case, *S-mask*s of all siTDs for this endpoint have a value of 10h (a one bit in micro-frame 4) and *C-mask* value of C3h (one-bits in micro-frames 0,1, 6 and 7). Additionally, sofware ensures that the *Back Pointer*

field of each siTD references the appropriate siTD data structure (and the *Back Pointer T-bits* are set to zero).

The initial *SplitXState* of the first siTD is **Do Start Split**. The host controller will visit the first siTD eight times during frame X. The C-mask bits in micro-frames 0 and 1 are ignored because the state is **Do Start Split**. During micro-frame 4, the host controller determines that it can run a start-split (and does) and changes *SplitXState* to **Do Complete Split**. During micro-frames 6 and 7, the host controller executes complete-splits. Notice the siTD for frame X+1 has it's *SplitXState* initialized to **Do Complete Split**. As the host controller continues to traverse the schedule during *H-Frame* X+1, it will visit the second siTD eight times. During micro-frames 0 and 1 it will detect that it must execute complete-splits.

During *H-Frame* X+1, micro-frame 0, the host controller detects that $siTD_{X+1}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+1}$ and fetches $siTD_X$. It executes the complete split transaction using the transaction state of $siTD_X$. If the $siTD_X$ split transaction is complete, siTD's *Active* bit is set to zero and results written back to $siTD_X$. The host controller retains the fact that $siTD_X$ is retired and transitions the *SplitXState* in the $siTD_{X+1}$ to **Do Start Split.** At this point, the host controller is prepared to execute the start-split for $siTD_{X+1}$ when it reaches micro-frame 4. If the split-transaction completes early (transaction-complete is defined in Section Periodic Isochronous—Do Complete Split), i.e. before all the scheduled complete-splits have been executed, the host controller will transition $siTD_X$*.SplitXState* to **Do Start Split** early and naturally skip the remaining scheduled complete-split transactions. For this example, $siTD_{X+1}$ does not receive a DATA0 response until *H-Frame* X+2, micro-frame 1.

During *H-Frame X+2,* micro-frame 0, the host controller detects that $siTD_{X+2}$'s *Back Pointer.T-bit* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. As described above, it executes another split transaction, receives an MDATA response, updates the transfer state, but does not modify the *Active* bit. The host controller returns to the context of $siTD_{X+2}$, and traverses it's next pointer without any state change updates to $siTD_{X+2}$. S

During *H-Frame X+2*, micro-frame 1, the host controller detects $siTD_{X+2}$'s *S-mask[0]* is a zero, saves the state of $siTD_{X+2}$ and fetches $siTD_{X+1}$. It executes another complete-split transaction, receives a DATA0 response, updates the transfer state and sets the *Active* bit to a zero. It returns to the state of $siTD_{X+2}$ and changes its *SplitXState* to **Do Start Split**. At this point, the host controller is prepared to execute start-splits for $siTD_{X+2}$ when it reaches micro-frame 4. <TBD… describe how software detects that there was missing micro-frames (don't think we care about missing out micro-frames. There is enough residual state to identify than not all transactions were executed.).

60.4.4.13 Host Controller Pause

When the host controller's *HCHalted* bit in the USBSTS register is a zero, the host controller is sending SOF (Start OF Frame) packets down all enabled ports. When the schedules are enabled, the EHCI host controller will access the schedules in main memory each micro-frame. This constant pinging of main memory is known to create CPU power management problems for mobile systems. Specifically, mobile systems aggressively manage the state of the CPU, based on recent history usage. In the more aggressive power saving modes, the CPU can disable its caches. Current PC architectures assume that bus-master accesses to main memory must be cache-coherent. So, when bus masters are busy touching memory, the CPU power management software can detect this activity over time and inhibit the transition of the CPU into its lowest power savings mode. USB controllers are bus-masters and the frequency at which they

access their memory-based schedules keeps the CPU power management software from placing the CPU into its lowest power savings state.

USB Host controllers don't access main memory when they are suspended. However, there are a variety of reasons why placing the USB controllers into suspend won't work, but they are beyond the scope of this document. The base requirement is that the USB controller needs to be kept out of main memory, while at the same time, the USB bus is kept from going into suspend.

EHCI controllers provide a large-grained mechanism that can be manipulated by system software to change the memory access pattern of the host controller. System software can manipulate the schedule enable bits in the USBCMD register to turn on/off the scheduling traversal. A software heuristic can be applied to implement an on/off duty cycle that allows the USB to make reasonable progress and allow the CPU power management to get the CPU into its lowest power state. This method is not intended to be applied at all times to throttle USB, but should only be applied in very specific configurations and usage loads. For example, when only a keyboard or mouse is attached to the USB, the heuristic could detect times when the USB is attempting to move data only very infrequently and can adjust the duty cycle to allow the CPU to reach it's low power state for longer periods of time. Similarly, it could detect increases in the USB load and adjust the duty cycle appropriately, even to the point where the schedules are never disabled. The assumption here is that the USB is moving data and the CPU will be required to process the data streams.

It is suggested that in order to provide a complete solution for the system, the companion host controllers should also provide a similar method to allow system software to inhibit the companion host controller from accessing it's shared memory based data structures (schedule lists or otherwise).

60.4.4.14 Port Test Modes

EHCI host controllers must implement the port test modes **Test J_State**, **Test K_State**, **Test_Packet**, **Test Force_Enable**, and **Test SE0_NAK** as described in the USB Specification Revision 2.0. The system is only allowed to test ports that are owned by the EHCI controller (e.g. *CF-bit* is a one and *PortOwner* bit is a zero). System software is allowed to have at most one port in test mode at a time. Placing more than one port in test mode will yield undefined results. The required, per port test sequence is as follows (assuming the *CF-bit* in the CONFIGFLAG register is a one):

1. Disable the periodic and asynchronous schedules by setting the *Asynchronous Schedule Enable* and *Periodic Schedule Enable* bits in the USBCMD register to a zero.
2. Place all enabled root ports into the suspended state by setting the *Suspend* bit in each appropriate PORTSC register to a one.
3. Set the *Run/Stop* bit in the USBCMD register to a zero and wait for the *HCHalted* bit in the USBSTS register, to transition to a one. Note that an EHCI host controller implementation may optionally allow port testing with the *Run/Stop* bit set to a one. However, all host controllers must support port testing with *Run/Stop* set to a zero and *HCHalted* set to a one.
4. Set the *Port Test Control* field in the port under test PORTSC register to the value corresponding to the desired test mode. If the selected test is **Test_Force_Enable**, then the *Run/Stop* bit in the USBCMD register must then be transitioned back to one, in order to enable transmission of SOFs out of the port under test.
5. When the test is complete, system software must ensure the host controller is halted (*HCHalted* bit is a one) then it terminates and exits test mode by setting *HCReset* to a one.

60.4.4.15 Interrupts

The EHCI Host Controller hardware provides interrupt capability based on a number of sources. There are several general groups of interrupt sources, as follows:

- Interrupts as a result of executing transactions from the schedule (success and error conditions)
- Host controller events (Port change events, etc.)
- Host Controller error events

All transaction-based sources are maskable through the Host Controller's Interrupt Enable register (USBINTR, see Section USBINTR). Additionally, individual transfer descriptors can be marked to generate an interrupt on completion. This section describes each interrupt source and the processing that occurs in response to the interrupt.

During normal operation, interrupts may be immediate or deferred until the next interrupt threshold occurs. The interrupt threshold is a tunable parameter via the *Interrupt Threshold Control* field in the USBCMD register. The value of this register controls when the host controller will generate an interrupt on behalf of normal transaction execution. When a transaction completes during an interrupt interval period, the interrupt signaling the completion of the transfer will not occur until the interrupt threshold occurs. For example, the default value is eight microframes. This means that the host controller will not generate interrupts any more frequently than once every eight microframes.

See Section , Host System Error," for details about the effects of a host system error.

If an interrupt has been scheduled to be generated for the current interrupt threshold interval, the interrupt is not signaled until after the status for the last complete transaction in the interval has been written back to host memory. This may sometimes result in the interrupt not being signaled until the next interrupt threshold.

Initial interrupt processing is the same, regardless of the reason for the interrupt. When an interrupt is signaled by the hardware, CPU control is transferred to host controller's USB interrupt handler. The precise mechanism to accomplish the transfer is OS specific. For this discussion it is just assumed that control is received. When the interrupt handler receives control, its first action is to reads the USBSTS (USB Status Register). It then acknowledges the interrupt by clearing all of the interrupt status bits by writing ones to these bit positions. The handler then determines whether the interrupt is due to schedule processing or some other event. After acknowledging the interrupt, the handler (via an OS-specific mechanism), schedules a deferred procedure call (DPC) which will execute later. The DPC routine processes the results of the schedule execution. The precise mechanisms used are beyond the scope of this document.

Note: the host controller is not required to de-assert a currently active interrupt condition when software sets the interrupt enables (in the USBINR register, see Section USBINTR) to a zero. The only reliable method software should use for acknowledging an interrupt is by transitioning the appropriate status bits in the USBSTS register (Section USBSTS) from a one to a zero.

60.4.4.15.1 Transfer/Transaction Based Interrupts

These interrupt sources are associated with transfer and transaction progress. They are all dependent on the next interrupt threshold.

Transaction Error

A transaction error is any error that caused the host controller to think that the transfer did not complete successfully. Table 60-104 lists the events/responses that the host can observe as a result of a transaction. The effects of the error counter and interrupt status are summarized in the following paragraphs. Most of these errors set the *XactErr* status bit in the appropriate interface data structure.

There is a small set of protocol errors that relate only when executing a queue head and fit under the umbrella of a WRONG PID error that are significant to explicitly identify. When these errors occur, the *XactErr* status bit in the queue head is set and the *CErr* field is decremented. When the *PIDCode* indicates a SETUP, the following responses are protocol errors and result in *XactErr* bit being set to a one and the *CErr* field being decremented.

- *EPS* field indicates a high-speed device and it returns a Nak handshake to a SETUP.
- *EPS* field indicates a high-speed device and it returns a Nyet handshake to a SETUP.
- *EPS* field indicates a low- or full-speed device and the complete-split receives a Nak handshake.

Table 60-104. Summary of Transaction Errors

Event / Result	Queue Head/qTD/iTD/siTD Side-effects		USB Status Register (USBSTS)
	Cerr	Status Field	USBERRINT
CRC	-1	XactErr set to a one.	1[1]
Timeout	-1	XactErr set to a one.	1[1]
Bad PID[2]	-1	XactErr set to a one.	1[1]
Babble	N/A	Section Serial Bus Babble	1
Buffer Error	N/A	Section Data Buffer Error	

1 If it occurs in a queue head, *USBERRINT* is asserted only when *CErr* counts down from a one to a zero. In addition the queue is halted. See Section , Halting a Queue Head ."

2 The host controller received a response from the device, but it could not recognize the PID as a valid PID.

Serial Bus Babble

When a device transmits more data on the USB than the host controller is expecting for this transaction, it is defined to be babbling. In general, this is called a *Packet Babble*. When a device sends more data than the *Maximum Length* number of bytes, the host controller sets the *Babble Detected* bit to a one and halts the endpoint if it is using a queue head (see Section Section , Halting a Queue Head "). *Maximum Length* is defined as the minimum of *Total Bytes to Transfer* and *Maximum Packet Size*. The *CErr* field is not decremented for a packet babble condition (only applies to queue heads). A babble condition also exists if IN transaction is in progress at High-speed EOF2 point. This is called a frame babble. A frame babble condition is recorded into the appropriate schedule data structure. In addition, the host controller must disable the port to which the frame babble is detected.

The *USBERRINT* bit in the USBSTS register is set to a one and if the *USB Error Interrupt Enable* bit in the USBINTR register is a one, then a hardware interrupt is signaled to the system at the next interrupt

threshold. The host controller must never start an OUT transaction that will babble across a micro-frame EOF.

NOTE

When a host controller detects a data PID mismatch, it must either: disable the packet babble checking for the duration of the bus transaction or do packet babble checking based solely on *Maximum Packet Size*. The USB core specification defines the requirements on a data receiver when it receives a data PID mismatch (e.g. expects a DATA0 and gets a DATA1 or visa-versa). In summary, it must ignore the received data and respond with an ACK handshake, in order to advance the transmitter's data sequence.

The EHCI interface allows System software to provide buffers for a Control, Bulk or Interrupt IN endpoint that are not an even multiple of the maximum packet size specified by the device. Whenever a device misses an ACK for an IN endpoint, the host and device are out of synchronization with respect to the progress of the data transfer. The host controller may have advanced the transfer to a buffer that is less than maximum packet size. The device will re-send its maximum packet size data packet, with the original data PID, in response to the next IN token. In order to properly manage the bus protocol, the host controller must disable the packet babble check when it observes the data PID mismatch.

Data Buffer Error

This event indicates that an overrun of incoming data or a underrun of outgoing data has occurred for this transaction. This would generally be caused by the host controller not being able to access required data buffers in memory within necessary latency requirements. These conditions are not considered transaction errors, and do not effect the error count in the queue head. When these errors do occur, the host controller records the fact the error occurred by setting the *Data Buffer Error* bit in the queue head, iTD or siTD.

If the data buffer error occurs on a non-isochronous IN, the host controller will not issue a handshake to the endpoint. This will force the endpoint to resend the same data (and data toggle) in response to the next IN to the endpoint.

If the data buffer error occurs on an OUT, the host controller must corrupt the end of the packet so that it cannot be interpreted by the device as a good data packet. Simply truncating the packet is not considered acceptable. An acceptable implementation option is to 1's complement the CRC bytes and send them. There are other options suggested in the Transaction Translator section of the USB Specification Revision 2.0.

USB Interrupt (Interrupt on Completion (IOC))

Transfer Descriptors (iTDs, siTDs, and queue heads (qTDs)) contain a bit that can be set to cause an interrupt on their completion. The completion of the transfer associated with that schedule item causes the USB Interrupt (USBINT) bit in the USBSTS register to be set to a one. In addition, if a short packet is encountered on an IN transaction associated with a queue head, then this event also causes USBINT to be set to a one. If the USB Interrupt Enable bit in the USBINTR register is set to a one, a hardware interrupt

is signaled to the system at the next interrupt threshold. If the completion is because of errors, the *USBERRINT* bit in the USBSTS register is also set to a one.

Short Packet

Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer. Whenever a short packet completion occurs during a queue head execution, the *USBINT* bit in the USBSTS register is set to a one. If the *USB Interrupt Enable* bit is set in the USBINTR register, a hardware interrupt is signaled to the system at the next interrupt threshold.

60.4.4.15.2 Host Controller Event Interrupts

These interrupt sources are independent of the interrupt threshold (with the one exception being the Interrupt on Async Advance, see Section Interrupt on Async Advance).

Port Change Events

Port registers contain status and status change bits. When the status change bits are set to a one, the host controller sets the *Port Change Detect* bit in the USBSTS register to a one. If the *Port Change Interrupt Enable* bit in the USBINTR register is a one, then the host controller will issue a hardware interrupt. The port status change bits include:

- Connect Status Change
- Port Enable/Disable Change
- Over-current Change
- Force Port Resume

Frame List Rollover

This event indicates that the host controller has wrapped the frame list. The current programmed size of the frame list effects how often this interrupt occurs. If the frame list size is 1024, then the interrupt will occur every 1024 milliseconds, if it is 512, then it will occur every 512 milliseconds, etc. When a frame list rollover is detected, the host controller sets the *Frame List Rollover* bit in the USBSTS register to a one. If the *Frame List Rollover Enable* bit in the USBINTR register is set to a one, the host controller issues a hardware interrupt. This interrupt is not delayed to the next interrupt threshold.

Interrupt on Async Advance

This event is used for deterministic removal of queue heads from the asynchronous schedule. Whenever the host controller advances the on-chip context of the asynchronous schedule, it evaluates the value of the *Interrupt on Async Advance Doorbell* bit in the USBCMD register. If it is a one, it sets the *Interrupt on Async Advance* bit in the USBSTS register to a one. If the *Interrupt on Async Advance Enable* bit in the USBINTR register is a one, the host controller issues a hardware interrupt at the next interrupt threshold. A detailed explanation of this feature is described in Section Removing Queue Heads from Asynchronous Schedule .

Host System Error

The host controller is a bus master and any interaction between the host controller and the system may experience errors. The type of host error may be catastrophic to the host controller (such as a Master Abort) making it impossible for the host controller to continue in a coherent fashion. In the presence of non-catastrophic host errors, such as parity errors, the host controller could potentially continue operation. The recommended behavior for these types of errors is to escalate it to a catastrophic error and halt the host controller. Host-based error must result in the following actions:

- The *Run/Stop* bit in the USBCMD register is set to a zero.
- The following bits in the USBSTS register are set:
  - *Host System Error* bit is to a one.
  - *HCHalted* bit is set to a one.
- If the *Host System Error Enable* bit in the USBINTR register is a one, then the host controller will issue a hardware interrupt. This interrupt is not delayed to the next interrupt threshold. Table 60-105 summarizes the required actions taken on the various host errors.

Table 60-105. Summary Behavior of EHCI Host Controller on Host System Errors

Cycle Type	Master Abort	Target Abort	Data Phase Parity
Frame list pointer fetch (read)	Fatal	Fatal	Fatal [o]
siTD fetch (read)	Fatal	Fatal	Fatal [o]
siTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
iTD fetch (read)	Fatal	Fatal	Fatal [o]
iTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
qTD fetch (read)	Fatal	Fatal	Fatal [o]
qHD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
Data write	Fatal [o]	Fatal [o]	Fatal [o]
Data read	Fatal	Fatal	Fatal [o]

[o] Potentially, a host controller implementation could continue operation without a halt. However, the recommended behavior is to halt the host controller.

NOTE

After a Host System Error, Software must reset the host controller via HCReset in the USBCMD register before re-initializing and restarting the host controller.

60.4.5 EHCI Deviation

For the purposes a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specificationEnhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. http://www.intel.com. Device operation and On-The-Go operation are not specified in the EHCI and thus the implementation supported

in this core is proprietary. The host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized as follows:

- Embedded Transaction Translator—Allows direct attachment of FS and LS devices in host mode without the need for a companion controller.
- Device operation—In host mode the device operational registers are generally disabled and thus device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface—This core does not a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.
- On-The-Go Operation—This design includes an On-The-Go controller for Port #1.

60.4.5.1 Embedded Transaction Translator Function

The ARC USB-HS OTG High-Speed USB On-The-Go OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction trasnslator. Although there is no separate Tranaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator fuction is an extension to EHCI interface, but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

60.4.5.1.1 Capability Registers

The following additions have been added to the capability registers to support the embedded Transaction Translator Function:

- N_TT added to HCSPARAMS—Host Control Structural Parameters
- N_PTT added toHCSPARAMS—Host Control Structural Parameters

60.4.5.1.2 Operational Registers

The following additions have been added to the operational registers to support the embedded TT:

- A new register.
- Addition of two-bit Port Speed (PSPD) to the PORTSCx register.

60.4.5.1.3 Discovery

In a standard EHCI controller design, the EHCI host controller driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation. The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (i.e. Chirp completes successfully).

Since this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result and the resulting port speed will be indicated by the PSPD field in PORTSCx.

Therefore, the standard EHCI host controller driver requires an alteration to handle directly connected Full and Low speed devices or hubs.

The change is a fundamental one in that is summarized in Table 60-106:

Table 60-106. Summary of EHCI

Standard EHCI	EHCI with embedded Transaction Translator
After port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After port enable bit is set following a connection and reset sequence, the device/hub speed is noted from PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, then port-level control is done using the Hub Class through the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr=X. **[where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (ie. Split target hub)]**	FS and LS device can be either downstream from a HS hub with HubAddr = X **[HubAddr > 0]** or directly attached **[where HubAddr = 0 and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (ie. Split target hub is the root hub)]**

60.4.5.1.4 Data Structures

The same data structures used for FS/LS transactions though a HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator. The following sequence demonstrates how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) – Async. (Bulk/Control Endpoints) Periodic (Interrupt)
   - Hub Address = 0
   - Transactions to direct attached device/hub.
     - QH.EPS = Port Speed
   - Transactions to a device downstream from direct attached FS hub.
     - QH.EPS = Downstream Device Speed

NOTE

When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

Maximum packet size must be less than or equal 64 or undefined behaviour may result.

2. siTD (for direct attach FS) – Periodic (ISO Endpoint)
   - All FS ISO transactions:
     - Hub Address = 0
     - siTD.EPS = 00 (full speed) (maximum packet size must less than or equal to 1023 or undefined behaviour may result.)

60.4.5.1.5 Operational Model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specificationUniversal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. http://www.usb.org) and for the EHCI controller moving packets between system memory and a USB-HS hub. Since the embedded Transaction Translator exists within the host controller there is no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

Micro- frame Pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the micro-frame pipeline implemented in the embedded Transaction Translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 will be ready to execute on the bus in B-frame 0.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators.

Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer can not babble through the SOF (start of B-frame 0.)

Split State Machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator. Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. Table 60-107 summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

Table 60-107. Summary of the Handshake Conditions

Condition	Emulate TT Response
Start-Split: All asynchronous buffers full.	NAK
Start-Split: All periodic buffers full.	ERR
Start-Split: Success for start of Async. Transaction.	ACK
Start-Split: Start Periodic Transaction.	No Handshake (Ok)

Table 60-107. Summary of the Handshake Conditions (continued)

Complete-Split: Failed to find transaction in queue.	Bus Time Out
Complete-Split: Transaction in Queue is Busy.	NYET
Complete-Split: Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

Note: The unshaded cells represent Start-Splits and the shaded cells represent Complete-Splits.

Asynchronous Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- USB 2.0 – 11.17.3—Sequencing is provided and a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.
- USB 2.0 – 11.17.4—Transaction tracking for 2 data pipes.
- USB 2.0 – 11.17.5—Clear_TT_Buffer capability provided though the use of the register.

Periodic Transaction Scheduling and Buffer Management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- USB 2.0 – 11.18.6.[1–2]
  - — Abort of pending start-splits
    - – EOF (and not started in micro-frames 6)
    - – Idle for more than 4 micro-frames
  - — Abort of pending complete-splits
    - – EOF
    - – Idle for more than 4 micro-frames
- USB 2.0 – 11.18.[7–8]
  - — Transaction tracking for up to 16 data pipes.
  - — Some applications may not require transaction tracking up to a maximum of 16 periodic data pipes. The option to limit tracking to only 4 periodic data pipes is chosen by changing the

configuration constant VUSB_HS_TT_PERIODIC_CONTEXTS to 4. The result is a significant gate count savings to the core given the limitations implied.

CAUTION

Limiting the number of tracking pipes in the EMBedded –TT to four (4) will impose the restriction that no more than 4 periodic transactions (INTERRUPT/ISOCHRONOUS) can be scheduled through the embedded-tt per frame. the number 16 was chosen in the USB specification because it is sufficient to ensure that the high-speed to full-speed periodic pipeline can remain full. keeping the pipeline full puts no constraint on the number of periodic transactions that can be scheduled in a frame and the only limit becomes the flight time of the packets on the bus.

— Complete-split transaction searching.

NOTE

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

Multiple Transaction Translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N_TT field in the HCSPARAMS—EHCI Compliant with extensions register.

60.4.5.2 Device Operation

The coexistence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

60.4.5.3 USBMODE Register

Given that the dual-role controller is initialized in neither host nor device mode, the USBMODE register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

60.4.5.3.1 Non-Zero Fields the register file.

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in device mode, the following must be adhered to:

- Write operations to all EHCI reserved fields (some of which are device fields) with the operation registers should always be written to zero. This is an EHCI requirement of the device controller driver that must be adhered to.
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields) because fields that are used exclusive for device are undefined in host mode .

60.4.5.3.2 SOF Interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for host mode. EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See USBSTS and USBINTR registers.

60.4.5.4 Embedded Design Interface

This is an Embedded USB Host Controller as defined by the EHCI specification and thus does not implement the PCI configuration registers.

60.4.5.4.1 Frame Adjust Register

Given that the optional PCI configuration registers are not included in this implementation, there is no corresponding bit level timing adjustments like is provided by the Frame Adjust register in the PCI configuration registers. Starts of micro-frames are timed precisely to 125 us using the transceiver clock as a reference clock. i.e. 60 Mhz transceiver clock for 8-bit physical interfaces & full-speed serial interfaces or 30 Mhz transceiver clock for 16-bit physical interfaces.

60.4.5.5 Miscellaneous Variations from EHCI

60.4.5.5.1 Programmable Physical Interface Behaviour

This design supports multiple physical interfaces which can operate in differing modes when the core is configured with software programmable physical interface modes. Software programmability allows the selection of the Physical interface part during the board design phase instead of during the chip design phase. The control bits for selecting the Physical Interface operating mode have been added to the PORTSCx register providing a capability that is not defined by EHCI.

60.4.5.5.2 Discovery

Port Reset

The port connect methods specified by EHCI require setting the port reset bit in the PORTSCx register for a duration of 10ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a ‘1’ to the reset the device.
- Software shall write a ‘0’ to the reset the device after 10 ms.
  - This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a ‘0’ to the reset bit while a reset is in progress the write will simple be ignored and the reset will continue until completion.

- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

Port Speed Detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation which will re-assign the port owner for any device that does not connect at High-Speed, this host controller supports direct attach of non High-Speed devices. Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed – *This information is redundant with the 2-bit Port Speed indicator above.*

60.4.5.5.3 Port Test Mode

Port Test Control mode behaves fully as described in EHCI since the release of revision 3.2.1. In earlier product revisions, the test packet mode was not EHCI compatible. An alternate host controller driver procedure is no longer necessary or supported.

60.4.6 Device Data Structures

This section defines the interface data structures used to communicate control, status, and data between Device Controller Driver (DCD) Software and the Device Controller. The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of device Queue Heads and Transfer Descriptors.

NOTE

Software must ensure that no interface data structure reachable by the Device Controller spans a 4K-page boundary.

The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/writeable fields. The device controller must preserve the read-only fields on all data structure writes.

The ARC USB-HS OTG High-Speed USB On-The-Go core includes DCD Software called the USB 2.0 device API. The device API provides an easy to use Application Program Interface for developing device (peripheral) applications using the ARC USB-HS OTG High-Speed USB On-The-Go core. The device API incorporates and abstracts for the application developer all of the elements of the program interface.

Figure 60-88 shows the endpoint queue head organization.

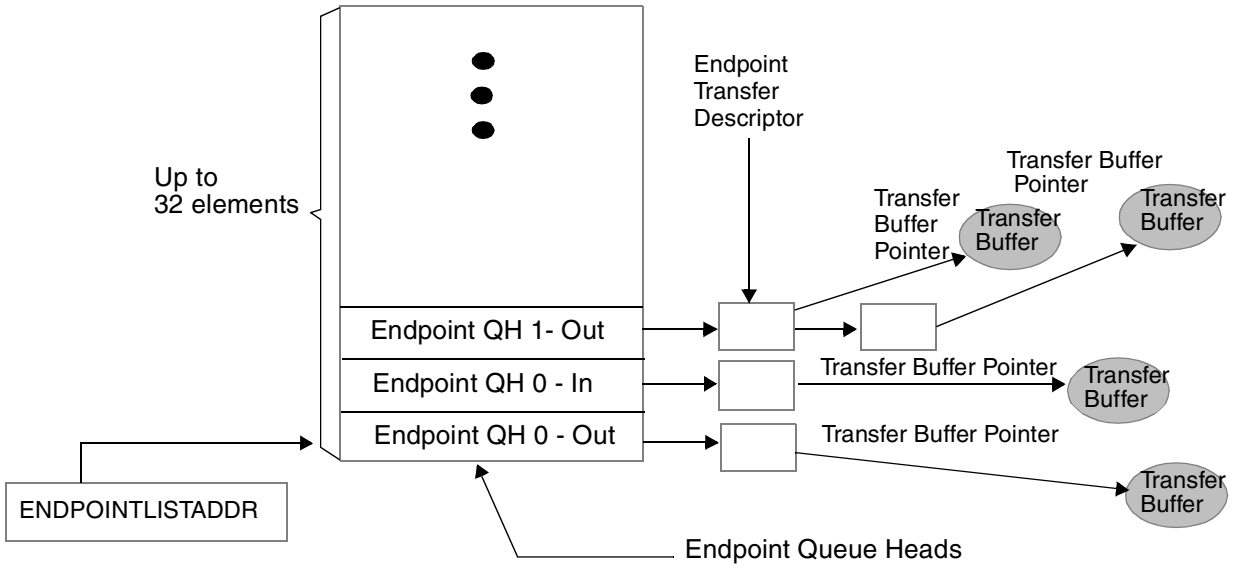


Figure 60-88. Endpoint Queue Head Organization

Device queue heads are arranged in an array in a continuous area of memory pointed to by the *ENDPOINTLISTADDR* pointer. The even –numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

NOTE

The Endpoint Queue Head List must be aligned to a 2k boundary.

60.4.6.1 Endpoint Queue Head (dQH)

The device Endpoint Queue Head (dQH) is where all transfers are managed. The dQH is a 48-byte data structure, but must be aligned on 64-byte boundaries. During priming of an endpoint, the dTD (device transfer descriptor) is copied into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, the dTD status DWord is updated in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH is used as a staging area for the dTD so that the Device Controller can access needed information with little minimal latency.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Mult		zlt		0				Maximum Packet Length								ios								0							
Current dTD Pointer																														0	

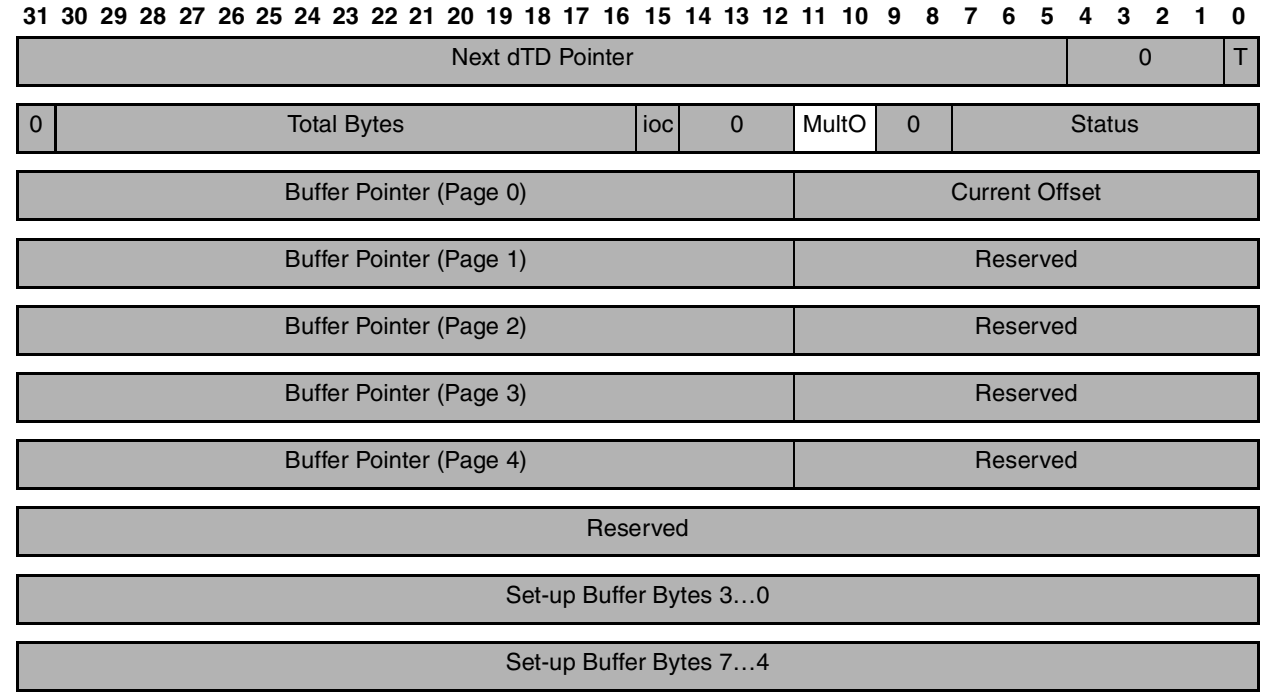


Figure 60-89. Endpoint Queue Head (dQH)

60.4.6.1.1 Endpoint Capabilities/Characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not attempt to modify this information while the corresponding endpoint is enabled.

Table 60-108 shows the endpoint capabilities and characteristics.

Table 60-108. Endpoint Capabilities/Characteristics

Bit	Description
31–30	**Mult.** This field is used to indicate the number of packets executed per transaction description as given by the following: 00 – Execute N Transactions as demonstrated by the USB variable length packet protocol where N is computed using the Maximum Packet Length (dQH) and the Total Bytes field (dTD) 01 – Execute 1 Transaction. 10 – Execute 2 Transactions. 11 – Execute 3 Transactions. Note: Non-ISO endpoints must set Mult=”00”. Note: ISO endpoints must set Mult=”01”, “10”, or “11” as needed.
29	**Zero Length Termination Select.** This bit is used to indicate when a zero length packet is used to terminate transfers where to total transfer length is a multiple . This bit is not relevant for Isochronous 0 – Enable zero length packet to terminate transfers equal to a multiple of the Maximum Packet Length. (default). 1 – Disable the zero length packet on transfers that are equal in length to a multiple Maximum Packet Length.
28–27	**Reserved.** These bit reserved for future use and should be set to zero.

Table 60-108. Endpoint Capabilities/Characteristics (continued)

26–16	**Maximum Packet Length.** This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 0x400 (1024).
15	**Interrupt On Setup (IOS).** This bit is used on control type endpoints to indicate if USBINT is set in response to a setup being received.
14–0	**Reserved.** Bits reserved for future use and should be set to zero.

60.4.6.1.2 Transfer Overlay

The seven DWords in the overlay area represent a transaction working space for the device controller. The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until a transfer is expired, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue.

See dTD for a description of the overlay fields.

60.4.6.1.3 Current dTD Pointer

The current dTD pointer is used by the device controller to locate the transfer in progress. This word is for Device Controller (hardware) use only and should not be modified by DCD software.

Table 60-109. Next dTD Pointer

Bit	Description
31–5	**Current dTD.** This field is a pointer to the dTD that is represented in the transfer overlay area. This field will be modified by the Device Controller to next dTD pointer during endpoint priming or queue advance.
4–0	**Reserved.** Bit reserved for future use and should be set to zero.

60.4.6.1.4 Set-up Buffer

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID.

NOTE

Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head is used for receiving setup data packets.

Table 60-110. Multiple Mode Control (HCCPARAMS)

DWord	Bits	Description
1	31–0	**Setup Buffer 0.** This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller to be read by software.
2	31–0	**Setup Buffer 1.** This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller to be read by software.

60.4.6.2 Endpoint Transfer Descriptor (dTD)

The dTD describes to the device controller the location and quantity of data to be sent/received for given transfer. The DCD should not attempt to modify any field in an active dTD except the Next Like Pointer, which should only be modified as described in Section 60.4.7.5, Managing Transfers with Transfer Descriptors.”

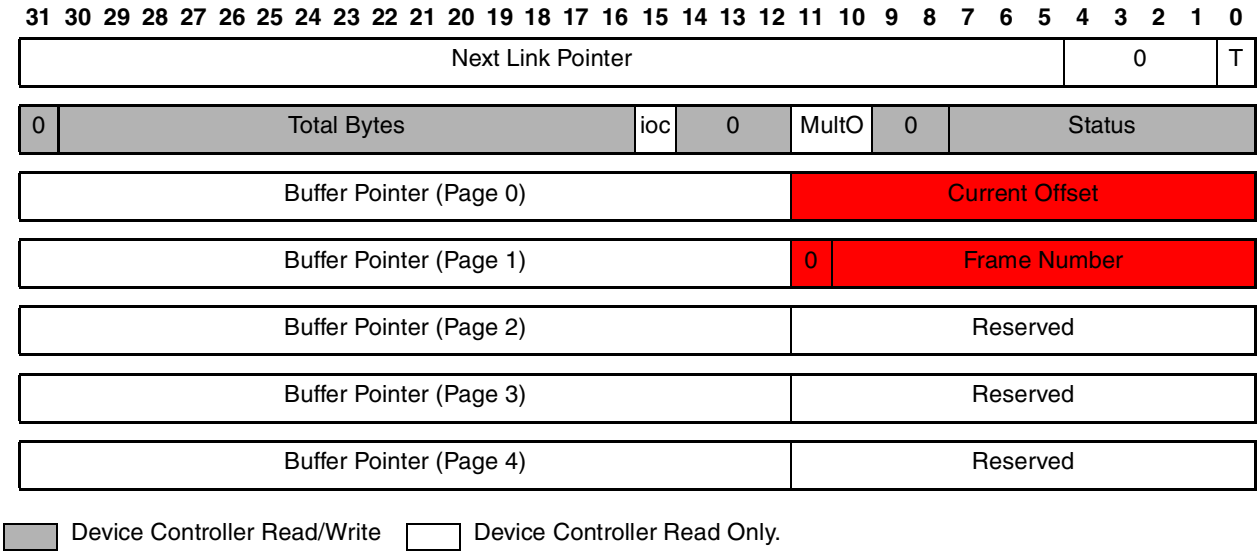


Figure 60-90. Endpoint Transfer Descriptor (dTD)

Table 60-111. Next dTD Pointer

Bit	Description
31–5	**Next Transfer Element Pointer.** This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4–1	**Reserved.** Bits reserved for future use and should be set to zero.
0	**Terminate (T).** 1=pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Device Controller that there are no more valid entries in the queue.

Table 60-112. dTD Token

Bit	Description
31	**Reserved.** Bit reserved for future use and should be set to zero.
30–16	**Total Bytes.** This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction and only on the successful completion of the transaction. The maximum value software may store in the field is 5 × 4K(5000H). This is the maximum number of bytes 5 page pointers can access. Although it is possible to create a transfer up to 20K this assumes the 1st offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. **Therefore, the maximum recommended transfer is 16K(4000H).** If the value of the field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the device controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for IN transfers that Total Bytes To Transfer be an even multiple of *Maximum Packet Length*. If software builds such a transfer descriptor for an IN transfer, the last transaction will always be less that *Maximum Packet Length*.
15	**Interrupt On Complete (IOC).** This bit is used to indicate if USBINT is to be set in response to device controller being finished with this dTD.
14–12	**Reserved**. Bits reserved for future use and should be set to zero.
11–10	**Multiplier Override (MultO).** This field can be used for transmit ISO's (ie. ISO-IN) to override the multiplier in the QH. This field must be zero for all packet types that are not transmit-ISO. Example: if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 0 [default] Three packets are sent: {Data2(8); Data1(7); Data0(0)} if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 2 Two packets are sent: {Data1(8); Data0(7)} For maximal efficiency, software should compute MultO = greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes = 0; then MultO should be 1. Note: Non-ISO and Non-TX endpoints must set MultO="00".
9–8	**Reserved**. Bits reserved for future use and should be set to zero.
7–0	**Status.** This field is used by the Device Controller to communicate individual command execution states back to the Device Controller software. This field contains the status of the last transaction performed on this qTD. The bit encodings are: Bit Status Field Description 7 Active. 6 Halted. 5 Data Buffer Error. 3 Transaction Error. 4,2,0 Reserved.

Table 60-113. dTD Buffer Page Pointer List

Bit	Description
31–12	**Buffer Pointer.** Selects the page offset in memory for the packet buffer. Non virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
0;11–0	**Current Offset.** Offset into the 4kb buffer where the packet is to begin.
1;10–0	**Frame Number.** Written by the device controller to indicate the frame number in which a packet finishes. This is typically be used to correlate relative completion times of packets on an ISO endpoint.

60.4.7 Device Operational Model

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus. Using a set of linked list **transfer descriptors**, pointed to by a **queue head**, the device controller will perform the data transfers. The following sections explain the use of the device controller from the device controller driver (DCD) point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

The ARC USB-HS OTG High-Speed USB On-The-Go is shipped with a DCD called the ARC USB-HS OTG High-Speed USB On-The-Go Device API. The ARC USB-HS OTG High-Speed USB On-The-Go Device API provides an easy to use application interface for developing USB device (peripheral) applications. The ARC USB-HS OTG High-Speed USB On-The-Go Device API incorporates and abstracts for the application developer all of the information contained in the device operational model. For more information on the ARC USB-HS OTG High-Speed USB On-The-Go Device API, refer to the "Software Design document for the Precise USB 2.0 Device API".

60.4.7.1 Device Controller Initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB D+ is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs. Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

1. Set Controller Mode in the USBMODE register to device mode.
2. Allocate and Initialize device queue heads in system memory. Note that Transitioning from host mode to device mode requires a device controller reset before modifying USBMODE.
   - Minimum: Initialize device queue heads 0 Tx and 0 Rx.
   - For information on device queue heads, refer to Section 60.4.6, Device Data Structures."

NOTE

All device queue heads must be initialized for control endpoints must be initialized before the endpoint is enabled. Non-Control device queue heads before the endpoint can be used.

3. Configure ENDPOINTLISTADDR Pointer.
   - For additional information on ENDPOINTLISTADDR, refer to the register table.
4. Enable the microprocessor interrupt associated with the ARC USB-HS OTG High-Speed USB On-The-Go core.
   - Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend.
   - For a list of available interrupts refer to the USBINTR and the USBSTS register tables.

5. Set Run/Stop bit to Run Mode.
   — After the Run bit is set, a device reset will occur. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the following Port State and Control section below.

NOTE

Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to initially prime Endpoint 0 because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework (Chapter 9) command set.

60.4.7.2 Port State and Control

From a chip or system reset, the device controller enters the *powered* state. A transition from the *powered* state to the *attach* state occurs when the Run/Stop bit is set to a '1'. After receiving a reset on the bus, the port enters the *defaultFS or defaultHS* state in accordance with the reset protocol described in Appendix C.2 of the USB Specification Rev. 2.0. Figure 60-91 depicts the state of a USB 2.0 device.

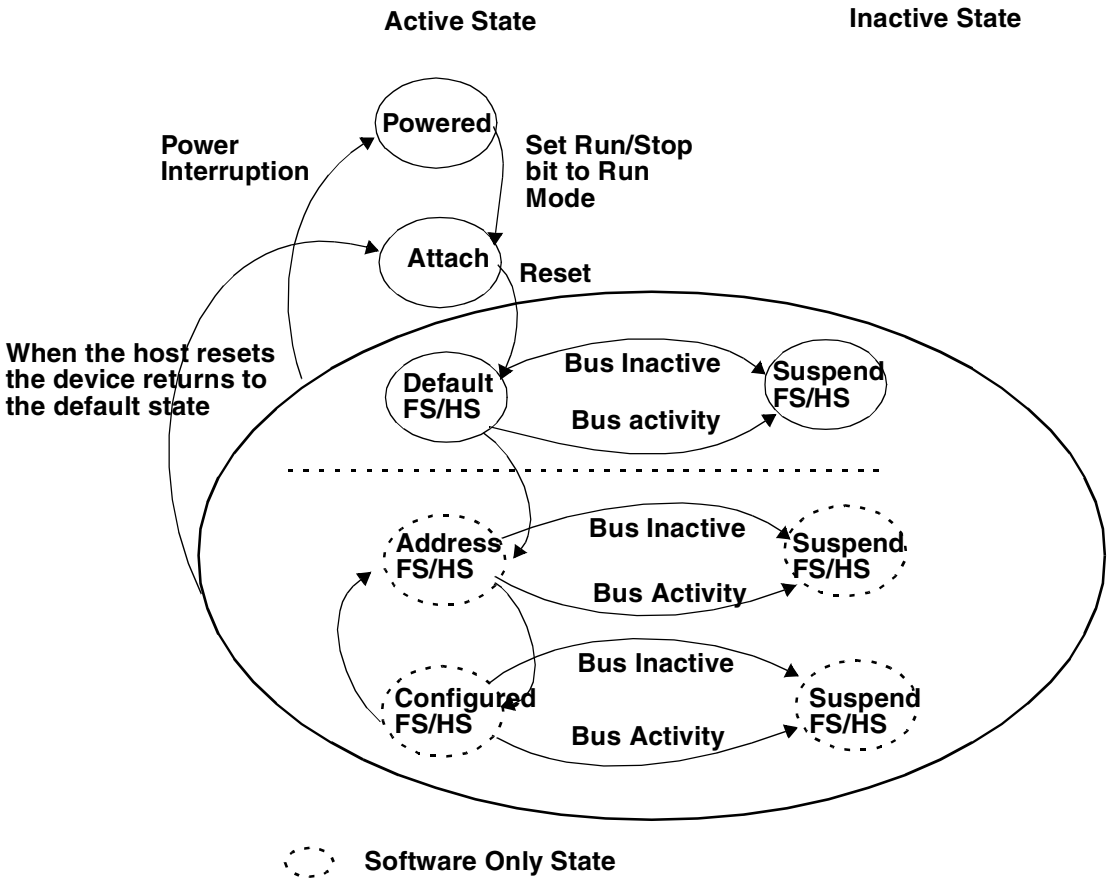


Figure 60-91. Device State Diagram

States *powered, attach, defaultFS/HS, suspendFS/HS* are implemented in the device controller and are communicated to the DCD using the following status bits as shown in Table 60-114:

Table 60-114. Device Controller State Information Bits

Bit	Register
DCSuspend	USBSTS
USB Reset Received	USBSTS
Port Change Detect	USBSTS
High-Speed Port	PORTSCx

It is the responsibility of the DCD to maintain a state variable to differentiate between the *DefaultFS/HS* state and the *Address/Configured* states. Change of state from *Default* to *Address* and the *Configured* states is part of the enumeration process described in the device framework section of the USB 2.0 Specification.

As a result of entering the *Address* state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the *Configured* indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the ENDPTCTRLx registers and initializing the associated queue heads.

60.4.7.2.1 Bus Reset

A bus reset is used by the host to initialize downstream devices. When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

Clear all setup token semaphores by reading the ENDPTSETUPSTAT register and writing the same value back to the ENDPTSETUPSTAT register.

Clear all the endpoint complete status bits by reading the ENDPTCOMPLETE register and writing the same value back to the ENDPTCOMPLETE register.

Cancel all primed status by waiting until all bits in the ENDPTPRIME are 0 and then writing 0xFFFFFFFF to ENDPTFLUSH.

Read the reset bit in the PORTSCx register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare.)

A hardware reset can be performed by writing a one to the device controller reset bit in the USBCMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.

Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated.

At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.

After a Port Change Detect, the device has reached the default state and the DCD can read the PORTSCx to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the USB Chapter 9 - Device Framework.

NOTE

The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device.

In some applications, it may not be possible to enable one or more pipes while in FS mode. *Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.*

60.4.7.2.2 Suspend

This section provides a general description of suspend and details about the operational mode.

Suspend Description

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither. Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wakeup. The ability of a device to signal remote wakeup is optional. If the USB device is capable of remote wakeup signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wakeup signaling must be disabled.

Suspend Operational Model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming *DC Suspend Interrupt* is enabled). When the *DCSuspend* bit in the PORTSCx is set to a ‘1’, the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation.

Information on the bus power limits in suspend state can be found in USB 2.0 specification.

NOTE

Review system level clocking issues defined in section (Ref: Signals-Clocking) for the clocking requirements of a suspended device controller.

60.4.7.2.3 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port. Resume signaling is sent upstream by writing a '1' to the Resume bit in the in the PORTSCx while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

NOTE

Before resume signaling can be used, the host must enable it by using the Set Feature command defined in the device framework chapter of the USB 2.0 Specification.

60.4.7.2.4 Port Test Modes

Contact ARC International for port test mode capabilities.

60.4.7.2.5 Managing Endpoints

The USB 2.0 specification defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device. The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a *control* type data channel used for device discovery and enumeration. Other types of endpoints support by USB include *bulk, interrupt*, and *isochronous*. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the USB 2.0 specification.

The ARC USB-HS OTG High-Speed USB On-The-Go device controller hardware supports up to the USB 2.0 maximum of 32 endpoint specified numbers. Each additional endpoint beyond the required endpoint position adds additional hardware logic. The maximum number of endpoint numbers available to the DCD is configured at hardware synthesis timer. After synthesis, the DCD can enable, disable and configure endpoint type up to the maximum selected during synthesis.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a *queue head* allocated in memory. If the maximum of 16 endpoint numbers, one for each endpoint direction are being used by the device controller, then 32 *queue heads* are required. The operation of an endpoint and use of *queue heads* are described later in this document.

60.4.7.2.6 Endpoint Initialization

After hardware reset, all endpoints except endpoint zero are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the ENDPTCTRLx register. Each 32-bit ENDPTCTRLx is split into an upper and lower half. The lower half of ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the ENDPTCTRLx register otherwise the behavior is undefined.

Table 60-115 shows how to construct a configuration word for endpoint initialization.

Table 60-115. Device Controller Endpoint Initialization

Field	Value
Data Toggle Reset	'1'
Data Toggle Inhibit	'0'
Endpoint Type	"00" – Control "01" – Isochronous "10" – Bulk "11" – Interrupt
Endpoint Stall	'0'

60.4.7.2.7 Stalling

There are two occasions where the device controller may need to return to the host a STALL

The first occasion is the **functional stall**, which is a condition set by the DCD as described in the USB 2.0 device framework. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.

A **protocol stall**, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

NOTE

Any write to the ENDPTCTRLx register during operational mode must preserve the endpoint type field (i.e. perform a read-modify-write).

Table 60-116. Device Controller Stall Response Matrix

USB Packet	Endpoint Stall Bit.	Effect on STALL bit.	USB Response
SETUP packet received by a non-control endpoint.	N/A	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'1'	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'0'	None.	ACK/ NAK/ NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	'1'	None	STALL
IN/OUT/PING packet received by a control endpoint.	'0'	None.	ACK/ NAK/ NYET

60.4.7.2.8 Data Toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe. For more information on data toggle, refer to the USB 2.0 specification.

Data Toggle Reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a '1' to the data toggle reset bit in the ENDPTCTRLx register. This should only be necessary when configuring/initializing an endpoint or returning from a STALL condition.

Data Toggle Inhibit

NOTE

This feature is for test purposes only and should never be used during normal device controller operation.

Setting the *data toggle Inhibit bit* active ('1') causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state.

In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

60.4.7.3 Operational Model For Packet Transfers

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification. At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were architected so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1 transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that can not be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, then we can expect the host will send IN requests to that endpoint. This device controller is architected in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as "**priming**" the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be architected properly use priming. Further, note that the term "**flushing**" is used to describe the action of clearing a packet that was queued for execution.

Priming Transmit Endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH). After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of primed is indicated in the ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB.

Since only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. More information about FIFO sizing is presented in section .

Priming Receive Endpoints

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host.

Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

60.4.7.3.1 Interrupt/Bulk Endpoint Operational Model

The behaviors of the device controller for interrupt and bulk endpoints are identical. All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

```
N = INT(Number Of Bytes/Max. Packet Length) + 1
```

With Zero Length Termination (ZLT) = 1

```
N = MAXINT(Number Of Bytes/Max. Packet Length)
```

Table 60-117. Variable Length Transfer Protocol Example (ZLT = 0)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	—
512	256	3	256	256	0
512	512	2	512	0	—

Table 60-118. Variable Length Transfer Protocol Example (ZLT = 1)

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	—
512	256	2	256	256	—
512	512	1	512	—	—

NOTE

The MULT field in the dQH must be set to “00” for bulk, interrupt, and control endpoints.

TX-dTD is complete when all packets described dTD were successfully transmitted. The total bytes in dTD equals zero when this occurs.

RX-dTD is complete when the following occurs.

- All packets described in dTD were successfully received. Total bytes in dTD equals zero when this occurs.

- A short packet (number of bytes < maximum packet length) was received. This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). This is an error condition. The device controller discards the remaining packet and sets the Buffer Error bit in the dTD. In addition, the endpoint is flushed and the USBERR interrupt becomes active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction.

On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

NOTE

All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

Interrupt/Bulk Endpoint Bus Response Matrix

Table 60-119 shows the interrupt/bulk endpoint bus response matrix.

Table 60-119. Interrupt/Bulk Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error	Force Bit Stuff Error
NYET/ACK	NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.
SYSERR	System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

60.4.7.3.2 Control Endpoint Operation Model

This section discusses the setup phase, data phase, status phase, and control endpoint bus response matrix.

Setup Phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

In hardware versions 2.3 and later, the setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted be an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

- Setup Packet Handling (Pre-2.3 hardware)

  After receiving an interrupt and inspecting USBMODE to determine that a setup packet was received on a particular pipe. perform the following steps:

  — 1. Duplicate contents of dQH.SsetupBuffer into local software byte array.

  — 2. Write 1 to clear corresponding ENDPTSETUPSTAT bit and thereby disable Setup Lockout. (i.e. the Setup Lockout activates as soon as a setup arrives. By writing to the ENDPTSETUPSTAT, the device controller accepts new setup packets.)

  — 3. Process setup packet using local software byte array copy and execute status/handshake phases.

  NOTE

  After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed and deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

  To limit the exposure of setup packets to the setup lockout mechanism (if used), the DCD should designate the priority of responding to setup packets above responding to other packet completions.

- Setup Packet Handling (2.3 hardware and later)

  a) Disable Setup Lockout by writing ‘1’ to Setup Lockout Mode (SLOM) in USBMODE. (once at initialization). Setup lockout is not necessary when using the tripwire as described below. Not that leaving the setup lockout mode as 0 results in pre-2.3 hardware behavior.

- After receiving an interrupt and inspecting ENDPTSETUPSTAT to determine that a setup packet was received on a particular pipe:

  a) Write '1' to clear corresponding bit ENDPTSETUPSTAT.

  b) Write ‘1’ to Setup Tripwire (SUTW) in USBCMD register.

c) Duplicate contents of dQH.SetupBuffer into local software byte array.

d) Read Setup TripWire (SUTW) in USBCMD register. (if set - continue; if cleared - goto 2)

e) Clear Setup Tripwire (SUTW) in USBCMD register.

f) Process setup packet using local software byte array copy and execute status/handshake phases.

NOTE

After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

Data Phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the ENDPTPRIME register is zero and the associated bit in the ENDPTSTAT register is a one. If a prime fails, ie. The ENDPTPRIME bit goes to zero and the ENDPTSTAT bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status (ENDPTSTAT) to enforce data coherency with the setup packet.

NOTE

The MULT field in the dQH must be set to “00” for bulk, interrupt, and control endpoints.

Error handling of data phase packets is the same as bulk packets described previously.

Status Phase

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase. The DCD must also perform the same checks of the ENDPTSETUPSTAT as described above in the data phase.

Note: The MULT field in the dQH must be set to “00” for bulk, interrupt, and control endpoints.

Note: Error handling of data phase packets is the same as bulk packets described previously.

Control Endpoint Bus Response Matrix

Table 60-120 shows the device controller response to packets on a control endpoint according to the device controller state.

Table 60-120. Control Endpoint Bus Response Matrix

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR	
In	STALL	NAK	Transmit	BS Error	N/A	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error — Force Bit Stuff Error

NYET/ACK — NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR — System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

60.4.7.3.3 Isochronous Endpoint Operational Model

Isochronous endpoints are used for real-time scheduled delivery of data and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes. Real time delivery by the device controller will is accomplished by the following:

- Exactly MULT Packets per (micro)Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets and sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro)frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes

the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. Once an ISO transaction is started in a (micro)frame it will retire the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the *Transaction Error* bit and the data is stored as usual for the application software to sort out.

- TX Packet Retired
  - MULT counter reaches zero.
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT

NOTE

For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field in hardware versions 2.3 and later. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.

- RX Packet Retired:
  - MULT counter reaches zero.
  - Non-MDATA Data PID is received
    - Exit criteria are only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
  - Overflow Error:
    - Packet received is > maximum packet length. [*Buffer Error* bit is set]
    - Packet received exceeds total bytes allocated in dTD. [*Buffer Error* bit is set]
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT
  - CRC Error [*Transaction Error* bit is set

NOTE

For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

Isochronous Pipe Synchronization

When it is necessary to synchronize an isochroous data pipe to the host, the (micro)frame number (FRINDEX register) can be used as a marker. To cause a packet transfer to occur at a specific (micro)frame number [N], the DCD should interrupt on SOF during frame N-1. When the FRINDEX=N–1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro)frame N–1 so that the device controller will execute delivery during (micro)frame N.

CAUTION

Priming an endpoint towards the end of (micro)frame N-1 will not guarantee delivery in (micro)frame N. The delivery may actually occur in (micro)frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

Isochronous Endpoint Bus Response Matrix

Table 60-121 shows the isochronous endpoint bus response matrix.

Table 60-121. Isochronous Endpoint Bus Response Matrix

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL Packet	NULL Packet	Transmit	BS Error	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error Force Bit Stuff Error

NULL Packet Zero Length Packet

60.4.7.4 Managing Queue Heads

Figure 60-92 shows the endpoint queue head diagram.

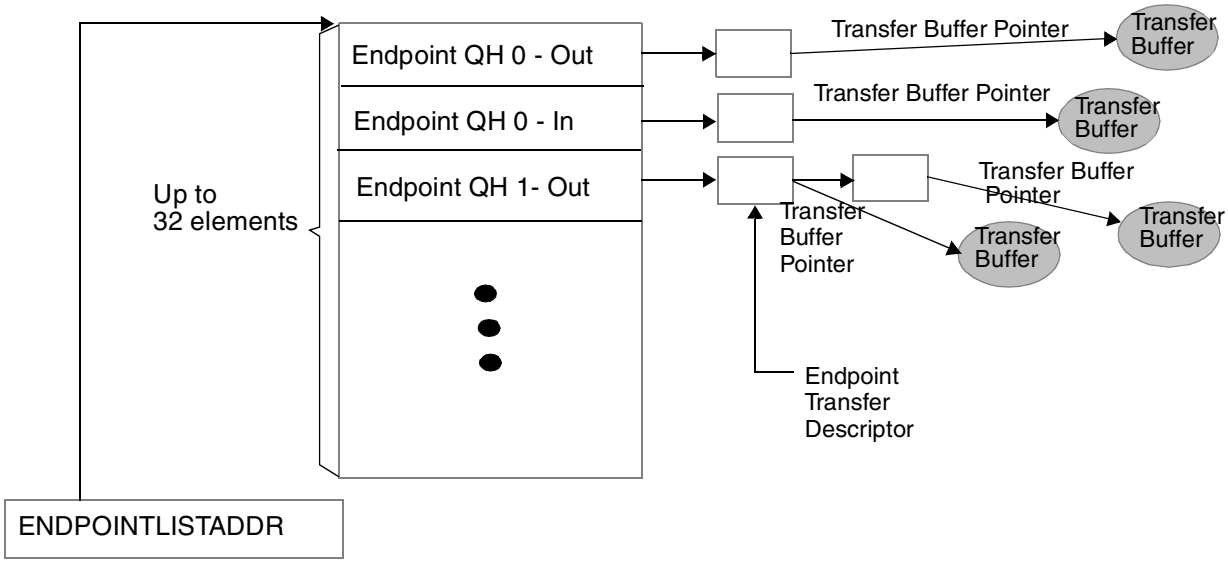


Figure 60-92. Endpoint Queue Head Diagram

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown in Figure 60-92. The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors since pointers will no longer exist within the queue head once the dTD is retired (see section Software Link Pointers).

In addition to the current and next pointers and the dTD overlay examined in section Operational Model For Packet Transfers, the dQH also contains the following parameters for the associated endpoint: Multipler, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

60.4.7.4.1 Queue Head Initialization

One pair of device queue heads must be initialized for each active endpoint. To initialize a device queue head, use the following sequence:

1. Write the wMaxPacketSize field as required by the USB Chapter 9 or application specific protocol.
2. Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1,2, or 3 as required bandwidth an in conjuction with the USB Chapter 9 protocol. *Note: In FS mode, the multiplier field can only be 1 for ISO endpoints.*
3. Write the next dTD Terminate bit field to "1".
4. Write the Active bit in the status field to "0".

5. Write the Halt bit in the status field to “0”.

NOTE

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTDs.

60.4.7.4.2 Operational Model For Setup Transfers

As discussed in section Control Endpoint Operation Model, setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a “1” to the corresponding bit in ENDPTSETUPSTAT.

NOTE

The acknowledge must occur before continuing to process the setup packet.

After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH – RX. Only the local software copy should be examined.

3. Check for pending data or status dTD’s from previous control transfers and flush if any exist as discussed in section Flushing/De-priming an Endpoint.
4. Decode setup packet and prepare data phase [optional] and status phase transfer as required by the USB Chapter 9 or application specific protocol.

NOTE

It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

60.4.7.5 Managing Transfers with Transfer Descriptors

This section explains how to use transfer descriptors to manage transfers.

60.4.7.5.1 Software Link Pointers

It is necessary for the DCD software to maintain head and tail pointers to the for the linked list of dTDs for each respective queue head. This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in the next section for managing dTD assumes the DCD can use reference the head and tail of the dTD linked list.

Figure 60-93 shows the software link pointers.

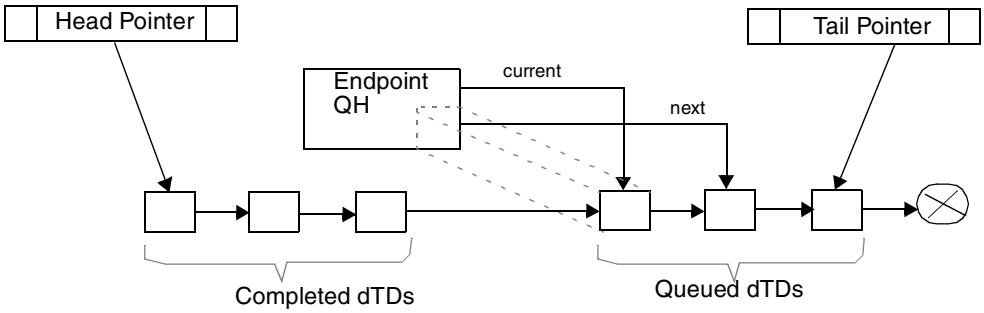


Figure 60-93. Software Link Pointers

NOTE

To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers but it still remains the responsibility of the DCD to maintain the pointers.

60.4.7.5.2 Building a Transfer Descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer. Use the following procedure for building dTDs.

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to “00000”

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to “1”.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to “1” and all remaining status bits set to “0”.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

60.4.7.5.3 Executing A Transfer Descriptor

To safely add a dTD, the DCD must be follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty:

— Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding)

- Case 1: Link list is empty

- 1. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
- 2. Clear active & halt bit in dQH (in case set from a previous error).
- 3. Prime endpoint by writing ‘1’ to correct bit position in ENDPTPRIME.

- Case 2: Link list is not empty
  - 1. Add dTD to end of linked list.
  - 2. Read correct prime bit in ENDPTPRIME – if ‘1’ DONE.
  - 3. Set ATDTW bit in USBCMD register to ‘1’.
  - 4. Read correct tatus bit in ENDPTSTAT. (store in tmp. variable for later)
  - 5. Read ATDTW bit in USBCMD register.
  - If ‘0’ goto 3.
  - If ‘1’ continue to 6.
  - 6. Write ATDTW bit in USBCMD register to ‘0’.
  - 7. If status bit read in (3) is ‘1’ DONE.
  - 8. If status bit read in (3) is ‘0’ then Goto Case 1: Step 1.

60.4.7.5.4 Transfer Completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

CAUTION

Multiple dTDs can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0
- Halted = 0
- Transaction Error = 0
- Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the Device Error Matrix.

In addition to checking the status bit the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is by decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

60.4.7.5.5 Flushing/De-priming an Endpoint

It is necessary for the DCD to flush to deprime one more endpoint on a USB device reset or during a broken control transfer. There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a '1' to the corresponding bit(s) in ENDPTFLUSH.
2. Wait until all bits in ENDPTFLUSH are '0'.
   - Software note: this operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.
3. Read ENDPTSTAT to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now '0'. If the corresponding bits are '1' after step #2 has finished, then the flush failed as described in the following:
   - Explanation: In very rare cases, a packet is in progress to the particular endpoint when commanded flush using ENDPTFLUSH. A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush be repeating steps 1-3 until each endpoint is successfully flushed.

60.4.7.5.6 Device Error Matrix

Table 60-122 summarizes packet errors that are not automatically handled by the Device Controller.

Table 60-122. Device Error Matrix

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Overflow **	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

Notice that the device controller handles all errors on Bulk/Control/Interrupt Endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated.

Table 60-123 shows the error descriptions.

Table 60-123. Error Descriptions

Error	Description
Overflow	Number of bytes received exceeded max. packet size or total buffer length. This error also sets the halt bit in the dQH. If there are dTDs remaining in the linked list for the endpoint, those will not be executed.
ISO Packet Error	CRC Error on received ISO packet. Contents not guaranteed to be correct.
ISO Fulfillment Error	Hst failed to complete the number of packets defined in the dQH mult field within the given (micro)frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro)frame. During the "dead" (micro)frame, the Device Controller reports error on the pipe and primes for the following frame.

60.4.7.6 Servicing Interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

60.4.7.6.1 High-Frequency Interrupts

High-frequency interrupts in particular should be handed in the order shown in Table 60-124. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

Table 60-124. High-Frequency Interrupt Events

Execution Order	Interrupt	Action
1a	USB Interrupt[1]—**ENDPTSETUPSTATUS**	Copy contents of setup buffer and acknowledge setup packet (as indicated in section). Process setup packet according to USB 2.0 Chapter 9 or application specific protocol.
1b	USB Interrupt[1]—**ENDPTCOMPLETE**	Handle completion of dTD as indicated in section .
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

[1] It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

60.4.7.6.2 Low-Frequency Interrupts

The low frequency events can be handled in any order because they do not occur often in comparison to the high-frequency interrupts. Table 60-125 shows the low-frequency interrupt events.

Table 60-125. Low Frequency Interrupt Events

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

60.4.7.6.3 Error Interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine. Table 60-126 shows the error interrupt events.

Table 60-126. Error Interrupt Events

Interrupt	Action
USB Error Interrupt	This error is redundant because it combines **USB Interrupt** and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of **USB Interrupt (w/ ENDPTCOMPLETE).**
System Error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

Chapter 61
Video Processing Unit (VPU)

61.1 Introduction

The video processing unit (VPU) is the multimedia video processing module in the i.MX51. Figure 61-1 shows the VPU top-level diagram.

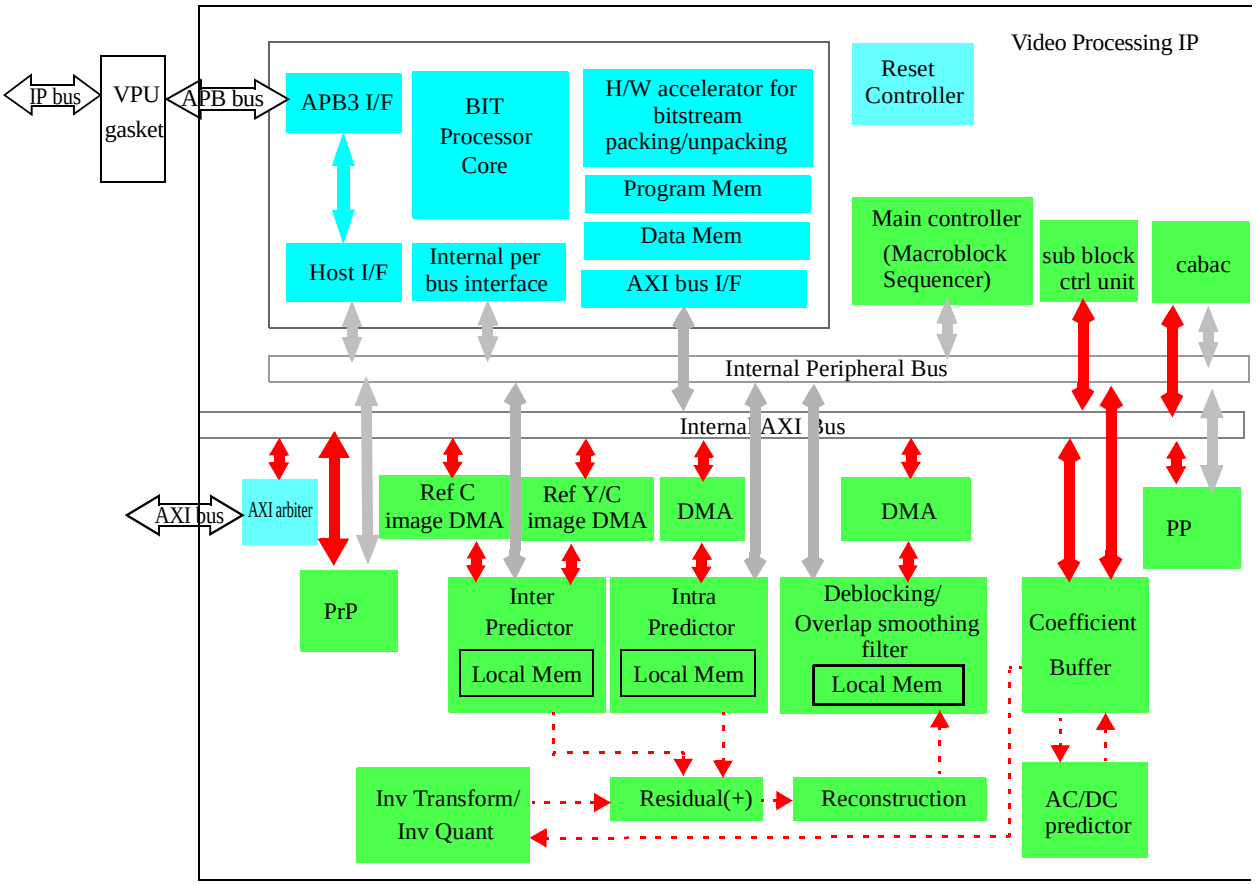


Figure 61-1. VPU Block Diagram

61.1.1 Overview

The video processing unit of the i.MX51 is a high performance, multistandard video processing unit that can perform H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG4 SP/ASP, Divx, RV8/9, and MPEG2 MP decoding up to 1920 × 1088 resolution. It supports multiple video codecs simultaneously.

The VPU has two bus interfaces: IP bus for register access controlling and AXI bus for data throughput. It makes use of external memory space for working buffer, frame buffer, parameter buffer, BIT processor program buffer and bitstream buffer. The VPU has 51 Kbyte internal memories to achieve high performance decoding.

The VPU's embedded BIT processor controls internal video processing subblocks and communicates with the host processor through host interface. The host CPU has a very low resource requirement for VPU support: it only needs to access VPU registers to initialize the VPU or to set parameters during frame gap. Applications are done through a VPU API called by host processor. A large part of the video codec (except BIT processor) is shared optimally, which achieves low power and a low gate count.

In combination with the processing capabilities of the image processing unit (IPU), the VPU can support high-quality video processing applications for the i.MX51.

61.1.2 Features

The main features of VPU are fully compliant with H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG-4 SP/ASP except GMC, Divx (Xvid), MPEG-1/2, and RV-8/9/10. The image size up to 1080p (1920 × 1088) is supported in decoding and SD (720 × 576) is supported for encoding. The VPU can support various error resilience tools and also supports multiple decoding and full duplex multi-party-call simultaneously.

The VPU is easy to integrate into the system because its interface is composed of general and simple AXI/APB. The VPU provides programmability, flexibility and ease of upgrading decoding/encoding or host interface because all the controls in decoding/encoding process and host interface are implemented as firmware in a programmable BIT processor.

Note that the capabilities of VPU are also depends on the system. For example, in the i.MX51, the maximum resolution of decoder should be 720p for all standards except MJPEG. This is mainly because of the busloading of the system.

Figure 61-2 shows a feature summary.

	Standard	Profile	Level	Resolution
Encoder	H.264	Baseline	3.0	720x576
	MPEG-4	SP		720x576
	H.263	Profile 3		720x576
	MJPEG	Baseline		8192x8192
Decoder	H.264	BP/MP/HP	4.0	1920x1088
	MPEG-4	ASP		1280x720
	H.263	Profile 3		1280x720
	VC-1	SP/MP/AP		1920x1088
	MPEG-2	Main	High	1920x1088
	Divx(Xvid)	Home theater		800x600
	RV	8/9/10		1920x1088
	MJPEG	Baseline		8192x8192

Figure 61-2. Feature Summary

The detailed features of the VPU are as follows:

- Multi-standard video codec
  - H.264/AVC decoder for baseline profile, main profile and high profile
  - VC-1 decoder for simple profile, main profile and advanced profile
  - MPEG-4 decoder for simple profile, advanced simple profile except GMC
  - H.263 decoder for baseline profile
  - Divx home theater decoder for profile (version 3.x, 4.x, 5.x, 6.x) and Xvid
  - MPEG-2 decoder for main profile @ high level
  - RV decoder for profile 8/9/10
  - H.264/AVC encoder for baseline profile
  - MPEG-4 encoder for simple profile
  - H.263 encoder for baseline profile
  - MJPEG encoder for baseline profile
  - Multiple codec: supports up to 4 decoding/encoding processes simultaneously, each process can have a different format.
- Other features
  - Supports rotating and mirroring simultaneously.
  - Built-in de-ringing filter.
  - Built-in de-blocking filter for MPEG-2/MPEG-4/Divx.
  - Simultaneous multi-stream and multi-standard processing capability.
  - Robust error detection/concelmenet.

- Programmability
  - Embeds 16-bit BIT processor that is dedicated to processing bitstream and controlling the hardware.
  - General purpose registers and interrupt generation for communication between host processor and VPU.

61.1.3 Modes of Operation

The VPU has two kinds of operating modes: normal operating mode and low power mode.

61.1.3.1 Normal Operating Mode

Normal operating mode is the video codec processing mode, which might be MPEG-4 encoding/decoding, Divx decoding, H.264 encoding/decoding, VC-1 decoding, or multiple bitstream encoding/decoding in multiple standards simultaneously. The host processor create and execute specified process by sending parameters and commands to the BIT processor through VPU API.

61.1.3.2 Low Power Mode

When VPU is in idle state, VPU interface signal vpu_busy is de-asserted and system can gate off its clock source according to this signal. If the clock source is gated off and system wants to wake it up, it reenables the clock and sends a command to VPU. If power is off when the system wants to wake it up, VPU has to be re-initialized. Refer to Section 61.2.1, Detailed Signal Descriptions," for a detailed description of the vpu_busy signal.

61.2 External Signal Description

The VPU has no external signals connecting off-chip. Table 61-1 lists the interface signals of VPU.

Table 61-1. Signal Interface List

Name	Function	Type	Reset	Comment
IP green-line clock and reset				
ipg_clk_s	IP green-line gated clock for the bus interface	input	—	—
ipg_hard_pos_async_reset_b	IP green-line hardware asynchronous reset	input	—	Active low
IP bus signals				
ips_addr[11:0]	IP bus address	input	—	—
ips_module_en	IP bus VPU module enable	input	—	Active high
ips_rwb	IP bus read/write (read if 1'b1)	input	—	—
ips_wdata[31:0]	IP bus write data	input	—	—
ips_rdata[31:0]	IP bus read data	output	32'h0	—
ips_xfr_err	IP bus transfer error	output	1'b0	Active high

Table 61-1. Signal Interface List (continued)

Name	Function	Type	Reset	Comment
ips_xfr_wait	IP bus transfer wait	output	1'b0	Active high
AXI signals - clock and reset				
aclk	AXI bus clock	input	—	—
areset_b	AXI bus reset signal	input	—	Active low
AXI first write address channel Interface				
o_pri_awid [3:0]	AXI write address ID	output	4'h0	—
o_pri_awaddr [31:0]	AXI write address	output	32'h0	—
o_pri_awlen [3:0]	AXI write burst length	output	4'h0	—
o_pri_awsize [2:0]	AXI write burst size	output	3'b011	always be 3'b011
o_pri_awburst [1:0]	AXI write burst type	output	2'b01	always be 2'b01
o_pri_awlock [1:0]	AXI write lock type	output	2'b00	always be 2'b00
o_pri_awcache [3:0]	AXI write cache type	output	4'h0	always be 4'h0
o_pri_awprot [2:0]	AXI write protection type	output	3'b010	always be 3'b010
o_pri_awvalid	AXI write address valid	output	1'b1	—
oi_pri_awready	AXI write address ready	input	—	—
AXI first write data channel Interface				
o_pri_wid [3:0]	AXI write data ID	output	4'h0	—
o_pri_wdata [63:0]	AXI write data	output	64'h0	—
o_pri_wstrb [7:0]	AXI write data stobes	output	8'hff	always be 8'hff
o_pri_wlast	AXI write last data	output	1'b0	—
o_pri_wvalid	AXI write data valid	output	1'b1	—
i_pri_wready	AXI write data ready	input	—	—
AXI first write response channel Interface				
i_pri_bid [3:0]	AXI write response ID	input	—	—
i_pri_bresp [1:0]	AXI write response	input	—	—
i_pri_bvalid	AXI write response valid	input	—	—
o_pri_bready	AXI write response ready	output	1'b1	—
AXI first read address channel Interface				
o_pri_arid [3:0]	AXI read address ID	output	4'h0	—
o_pri_araddr [31:0]	AXI read address	output	32'h0	—
o_pri_arlen [3:0]	AXI read burst length	output	4'h0	—
o_pri_arsize [2:0]	AXI read burst size	output	3'b011	always be 3'b011
o_pri_arburst [1:0]	AXI read burst type	output	2'b01	always be 2'b01

Table 61-1. Signal Interface List (continued)

Name	Function	Type	Reset	Comment
o_pri_arlock [1:0]	AXI read lock type	output	2'b00	always be 2'b00
o_pri_arcache [3:0]	AXI read cache type	output	4'h0	always be 4'h0
o_pri_arprot [2:0]	AXI read protection type	output	3'b010	always be 3'b010
o_pri_arvalid	AXI read address valid	output	1'b1	—
i_pri_arready	AXI read address ready	input	—	—
AXI first read data channel Interface				
i_pri_rid [3:0]	AXI read data ID	input	—	—
i_pri_rdata [63:0]	AXI read data	input	—	—
i_pri_rresp[1:0]	AXI read data response	input	—	—
i_pri_rlast	AXI read last data	input	—	—
i_pri_rvalid	AXI read data valid	input	—	—
o_pri_rready	AXI read data ready	output	1'b1	always be 1'b1
VPU core clock				
cclk	VPU core clock	input	—	—
creset_n	VPU core reset	input	—	—
IP Indigo Line IF				
ipi_int_vpu	signal to interrupt controller	output	1'b0	active high
Other control signal				
rclk	VPU reference clock	input	—	—
rreset_n	VPU reference reset	input	—	—
vpu_idle	VPU idle state signal	output	1'b1	—
vpu_underrun	VPU under running signal	output	1'b0	—

61.2.1 Detailed Signal Descriptions

All VPU input clock and reset sources come from the i.MX51 CCM module. The VPU does not do clock gating internally.

VPU uses one 64-bit AMBA3 AXI bus interface for data transfer from/to external memory. The VPU gasket module is responsible for transferring AMBA APB bus to IPS bus protocol.

The interrupt signal ipi_int_vpu is VPU module interrupt signal, active high. When VPU interrupt is enabled in system, its corresponding interrupt service routine is called when ipi_int_vpu signal raised. This signal is retained until host processor clear it. This signal is synchronized to the positive edge of the aclk.

The vpu_idle signal is used for VPU Clock gating control, active high. It indicates whether VPU is busy or idle. If vpu_idle is high, VPU is not running and the system can cut off the clock source. If VPU clock is off, wake up the VPU by turning on the clock and running the VPU through BIT processor command.

61.3 Memory Map and Register Definition

VPU registers are all 32-bit wide, and only support 32-bit aligned read/write operation. VPU registers are grouped into several regions corresponding to different decoding process step. They are used for decoding process configuration and control. They can only be accessed through IP bus interface.

61.3.1 Memory Map

Please note that there are some undefined address space in VPU memory map, any read/write accessing to this register address space is ignored in the VPU. Read accessing to write only register returns ZERO value. Write accessing to read only register is ignored.

The BIT processor registers' memory map in VPU is 0xBASE+0x0000~0xBASE+0x01D0. The BIT processor registers are divided into two categories.

- Address 0xBASE+0x0000~0xBASE+0x00FC (64 registers address space) are hardware registers. These registers have reset values and their functions are fixed (not configurable).
- Address 0xBASE+0x0100~0xBASE+0x01FC (64 registers address space) are software registers. They have no reset values and can be configured by internal BIT processor. Their definitions may change for different firmware version, so they are not provided here. This type of registers can be used as general parameter registers between host processor and BIT processor.
  - The first 32 parameter registers (address 0xBASE+0x0100~0xBASE+0x017C) are used as static parameters. Definition and functions of those registers are same for all kinds of run commands.
  - The second 32 parameter registers (address 0xBASE+0x0180~0xBASE+0x01DC) are used as temporal parameters. The definition and functions of those registers may differ in different run commands.

The memory map for the hardware registers of VPU is shown in Table 61-2.

Table 61-2. VPU Hardware Register Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x0000 (CodeRun)	BIT Processor run start	W	0x0000_0000	61.3.3.1/61-10
0xBASE+0x0004 (CodeDown)	BIT Boot Code Download Data register	W	0x0000_0000	61.3.3.2/61-10
0xBASE+0x0008 (HostIntReq)	Host Interrupt Request to BIT	W	0x0000_0000	61.3.3.3/61-11
0xBASE+0x000C (BitIntClear)	BIT Interrupt Clear	W	0x0000_0000	61.3.3.4/61-11
0xBASE+0x0010 (BitIntSts)	BIT Interrupt Status	R	0x0000_0000	**61.3.3.5/61-12**
0xBASE+0x0014 (BitCodeReset)	BIT Code Reset	W	0x0000_0000	**61.3.3.6/61-12**
0xBASE+0x0018 (BitCurPc)	BIT Current PC	R	0x0000_0000	**61.3.3.7/61-13**

Table 61-2. VPU Hardware Register Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE_0020 (BitCodecBusy)	BIT CODEC Busy	R/W	0x0000_0000	61.3.3.8/61-13
0xBASE+0x001C ~ 0xBASE+0x00FC	Reserved	—	—	—

Please refer to the VPU API document for descriptions on software registers access.

61.3.2 Register Summary

The conventions in Figure 61-3 and Figure 61-3 serve as a key for the register summary and individual register diagrams.

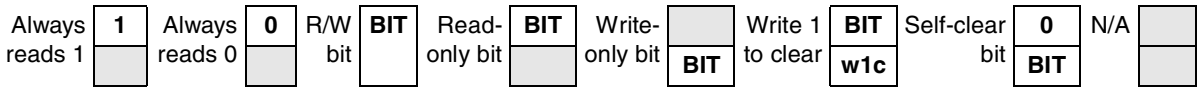


Figure 61-3. Key to Register Fields

Table 61-3 provides a key for register figures and tables and the register summary.

Table 61-3. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

The brief VPU hardware register summary is shown in Table 61-4.

Table 61-4. VPU Hardware Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x0000 (CodeRun)	R																
	W																
	R																
	W																CodeRun
0xBASE+0x0004 (CodeDown)	R																
	W				CodeAddr												
	R																
	W	CodeData															
0xBASE+0x0008 (HostIntReq)	R																
	W																
	R																
	W																IntReq
0xBASE+0x000C (BitIntClear)	R																
	W																
	R																
	W																IntClear
0xBASE+0x0010 (BitIntSts)	R																
	W																
	R																IntSts
	W																
0xBASE+0x0014 (BitCodeReset)	R																
	W																
	R																
	W																CodeReset
0xBASE+0x0018 (BitCurPc)	R																
	W																
	R			CurPc													
	W																

61.3.3 Register Descriptions

This section consists of VPU hardware register descriptions in address order. Each description includes a standard register diagram. A table containing the register bit and field function follows the register diagrams.

61.3.3.1 VPU Code Run Register (CodeRun)

See Figure 61-4 for illustration of valid bits in VPU Code Run Register and Table 61-5 for description of the bit fields in the register.

Address 0xBASE+0x0000 (CodeRun)

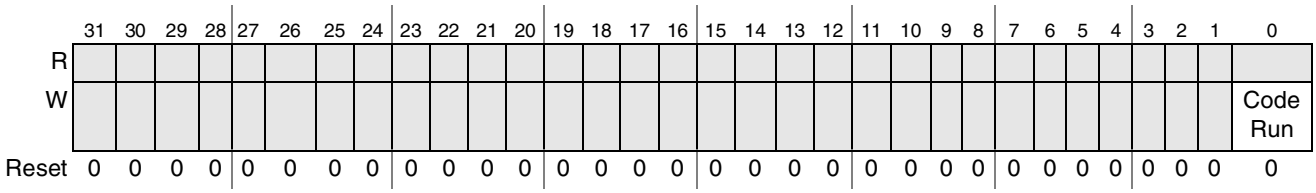


Figure 61-4. VPU Code Run Register

Table 61-5. VPU Code Run Register Field Descriptions

Field	Description
31–1	Reserved
0	CodeRun. BIT processor run start bit. 0 BIT Processor stop execution. 1 BIT Processor start execution.

61.3.3.2 VPU BIT Boot Code Download Data Register (CodeDown)

See Figure 61-5 for illustration of valid bits in VPU BIT Boot Code Download Data Register and Table 61-6 for description of the bit fields in the register.

Address 0xBASE+0x0004 (CodeDown)

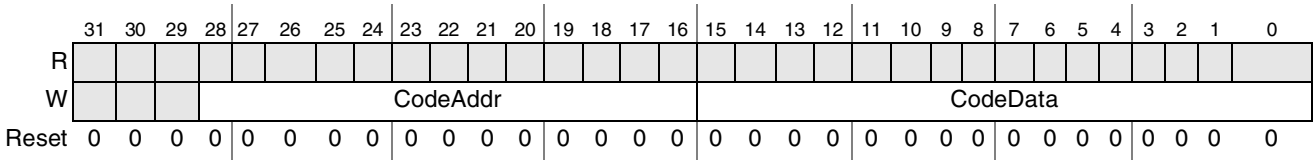


Figure 61-5. VPU BIT Boot Code Download Data Register

Table 61-6. VPU BIT Boot Code Download Data Register Field Descriptions

Field	Description
31–29	Reserved

Table 61-6. VPU BIT Boot Code Download Data Register Field Descriptions (continued)

Field	Description
28–16	CodeAddr[12:0] Download address of VPU BIT boot code, which is VPU internal address of BIT processor.
15–0	CodeData[15:0] Download data of VPU BIT boot code.

61.3.3.3 VPU Host Interrupt Request Register (HostIntReq)

See Figure 61-6 for illustration of valid bits in VPU Host Interrupt Request Register and Table 61-7 for description of the bit fields in the register.

Address 0xBASE+0x0008 (HostIntReq)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																IntReq
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 61-6. VPU Host Interrupt Request Register

Table 61-7. VPU Host Interrupt Request Register Field Descriptions

Field	Description
31–1	Reserved
0	IntReq. The host interrupt request bit. 0 No host interrupt is requested. 1 The host processor request interrupt to the BIT processor.

61.3.3.4 VPU BIT Interrupt Clear Register (BitIntClear)

See Figure 61-7 for illustration of valid bits in VPU BIT Interrupt Clear Register and Table 61-8 for description of the bit fields in the register.

Address 0xBASE+0x000C (BitIntClear)

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																IntClear
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 61-7. VPU BIT Interrupt Clear Register

Table 61-8. VPU BIT Interrupt Clear Register Field Descriptions

Field	Description
31–1	Reserved
0	IntClear. BIT interrupt clear bit. 0 No operation is issued. 1 Clear the BIT interrupt to the host.

61.3.3.5 VPU BIT Interrupt Status Register (BitIntSts)

See Figure 61-8 for illustration of valid bits in VPU BIT Interrupt Status Register and Table 61-9 for description of the bit fields in the register.

Address 0xBASE+0x0010 (BitIntSts)

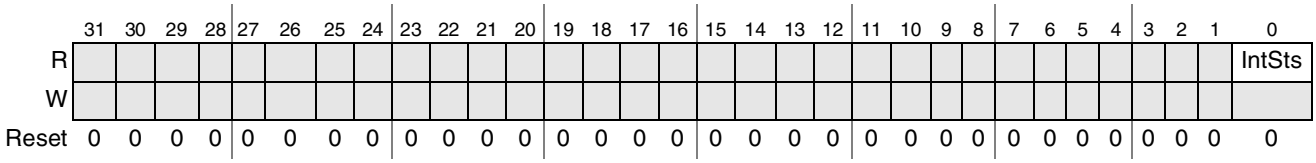


Figure 61-8. VPU BIT Interrupt Status Register

Table 61-9. VPU BIT Interrupt Status Register Field Descriptions

Field	Description
31–1	Reserved
0	IntSts. BIT interrupt status bit. 0 No BIT interrupt is asserted. 1 The BIT interrupt is asserted to the host. It is cleared when the host processor write “1” to BitIntClear register.

61.3.3.6 VPU BIT Code Reset Register (BitCodeReset)

See Figure 61-9 for illustration of valid bits in VPU BIT Code Reset Register and Table 61-10 for description of the bit fields in the register.

Address 0xBASE+0x0014 (BitCodeReset)

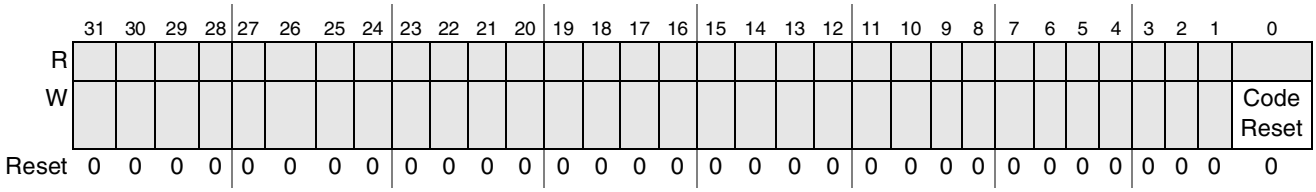


Figure 61-9. VPU BIT Code Reset Register

Table 61-10. VPU BIT Code Reset Register Field Descriptions

Field	Description
31–1	Reserved
0	CodeReset. BIT code reset bit. 0 No operation is issued. 1 The program counter of BIT processor is set to “0”, BIT processor restart at initial routine.

61.3.3.7 VPU BIT Current PC Register (BitCurPc)

See Figure 61-10 for illustration of valid bits in VPU BIT Current PC Register and Table 61-11 for description of the bit fields in the register.

Address 0xBASE+0x0018 (BitCurPc)

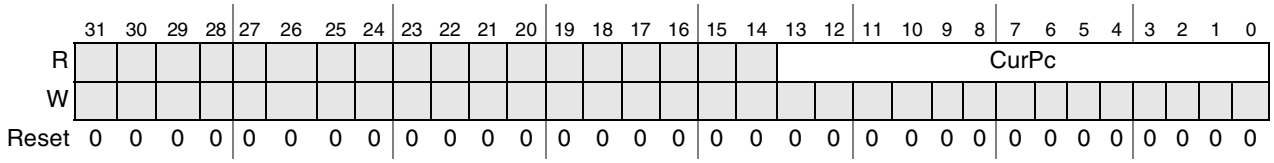


Figure 61-10. VPU BIT Current PC Register

Table 61-11. VPU BIT Current PC Register Field Descriptions

Field	Description
31–14	Reserved
13–0	CurPc[13:0]. BIT current PC value. Returns the current program counter of BIT processor by reading this register.

61.3.3.8 VPU BIT Busy Register (BitCodecBusy)

See Figure 61-10 for illustration of valid bits in VPU BIT Codec Busy Register and Table 61-11 for description of the bit fields in the register

Address 0xBASE+0x0020 (BitCodecBusy)

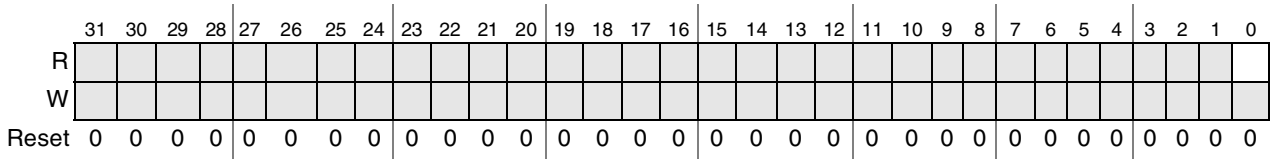


Figure 61-11. VPU BIT Busy Flag Register

Table 61-12.

field	Description
31–1	Reserved
0	Codec busy flag for Bit processor. BIT processor write “1”to this register when the processor is running.”0”means processor is waiting for a command. This value is connected to the o_vpu_idle.

61.4 Functional Description

VPU is a high-performance, multistandard video processing module in the i.MX51 that supports H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG4 SP/ASP, Divx and MPEG2 MP decoding up to 1920 × 1088 resolution.

61.4.1 VPU Architecture

The VPU is a high performance multi-standard video codec IP that can perform the H.264 BP/MP/HP, VC-1 SP/MP/AP, MPEG-4 SP/ASP, Divx, MPEG-1/2, RV-8/9/10, and MJPEG decoding and encoding. The VPU supports up to HD (1920 × 1088) decoding and SD (720 × 576) encoding. It can encode or decode multiple video clips with multiple standards simultaneously.

It connects with the system via the 32-bit AMBA 3 APB bus for system control and 64-bit AMBA3 AXI for data throughput, and takes advantage of on-chip memories to achieve high performance. The CodaHx14 has a 16-bit DSP, called as BIT processor. The BIT processor controls the internal video codec sub blocks and communicates with a host processor through the host interface. The required resource for the host CPU to control the VPU is very low, under 1 MIPS, because all functions such as rate control, FMO, ASO, video codec control, and error resilience are implemented in the BIT processor. The majority of the subblocks in the CodaHx14 is optimally shared, which achieves the ultra low power and low gate count.

VPU mainly includes two components: Video CODEC Processing IP and VPU gasket. Video CODEC Processing IP is the heart of the VPU. It is responsible for encoding/decoding. It mainly consists of an embedded 16-bit BIT processor, video codec hardware, and bus arbiter/interface. VPU gasket is in charge of converting AMBA APB3 bus to IP bus. Refer to Figure 61-1 for the block diagram of VPU. The following sections describe the main function of Video CODEC Processing IP components.

61.4.1.1 Embedded BIT processor

The embedded BIT processor is used for parsing or forming bitstreams. It includes some hardware accelerators to speed up bitstream processing. In addition to handling bitstreams, the BIT processor controls the video decoding hardware and communicates with host processor through IP bus and AXI bus interface.

61.4.1.2 Video CODEC Hardware

All video decoding processing, except handling coefficients for VLD, are implemented in hardware. The video codec hardware is designed to reduce logic gate count by sharing parts of sub-modules for multi-standard video decoding. It mainly includes the following hardware components.

61.4.1.2.1 Inter-Predictor

The Inter-Predictor module uses a reconstructed motion vector that represents the displacement between the block currently being decoded and the corresponding location in the reference frame, to calculate interpolated pixel data for motion compensation.

61.4.1.2.2 AC/DC and Intra-Predictor

There are two intra-prediction modules in the video decoding hardware. One is for the MPEG-4/H.263 P3/VC-1 AC/DC prediction and another for the H.264 intra-prediction. In case of VC-1 decoding, the hardware prediction mode decision is used. It brings high performance and low power consumption. The coefficient data of decoding is re-ordered automatically based on the detected prediction mode. For H.264 intra-prediction mode in decoding, hardware mode decision or software based made decision is used.

61.4.1.2.3 Inverse Transform/inverse Quantization

VPU has only one transform/quantization module. It operates in several types of inverse transform/quantization for MPEG-2, VC-1 and H.264. Each inverse transform/quantization for decoding uses different coefficients and different data processes, but it uses same control and data flow in many cases.

61.4.1.2.4 De-blocking/Overlap-Smoothing Filter

Deblocking filter removes blocking artifacts resulting from quantization and different motion vectors. The filter processing supports H.264 and VC-1, and operates within coding loop. Filtered frames are used as reference frames for motion compensation of subsequent coded frames. Overlap-smoothing filter supports VC-1 overlap-smoothing feature, and processed before deblocking filter. For H.264, H263 and VC-1, the de-blocking filter operates within decoding loop. Filtered frames are used as reference frames for motion compensation of subsequent coded frames.

61.4.1.2.5 Coefficient buffer interface

Coefficient buffer interface provides a channel for the BIT processor to send variable-length-decoded coefficients to video decoding hardware for decoding process. The coefficient buffer interface also performs reordering of coefficients based on the scan type.

61.4.1.2.6 Macroblock controller

VPU video decoding processing has a complex and large number of pipelines for high-performance. It’s not suitable to manage it wholly by the BIT processor. The macroblock controller is to control all sub modules of the video decoding hardware based on the configuration of pipelining by the BIT processor. The pipeline structure may change for different firmware running on the BIT processor. This scheme reduces the load on the BIT processor and guarantees the programmability of the IP.

Before decoding a macroblock, the BIT processor configures how the pipeline of the decoding is structured. If all processes are completed for decoding a macroblock, the macroblock controller indicates its completion. In summarization, the BIT processor configures which sub-modules are enabled for current macroblock processing, and the macroblock controller manages corresponding sub-modules based on the pipelining configuration.

Figure 61-12 shows the connectivity of macroblock controller.

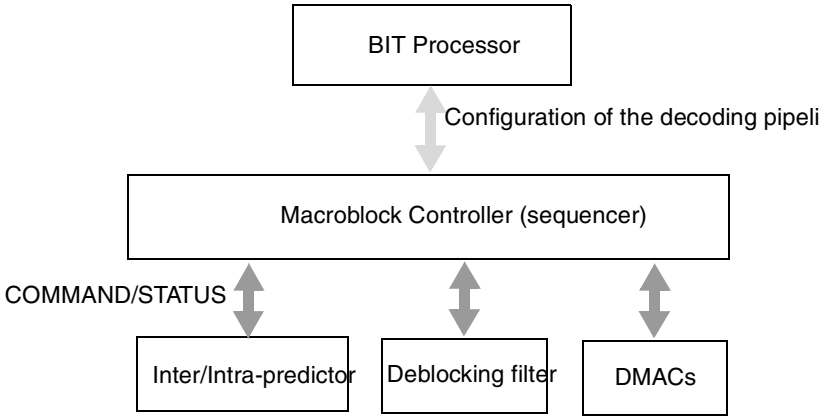


Figure 61-12. Connectivity of Macroblock Controller

61.4.1.2.7 Rotation/Mirroring

VPU supports rotation together with mirroring function for decoding output image to display. It is done by rotator/mirror module. The rotation/mirroring process in decoding process requires additional bandwidth because VPU has to re-use the un-rotated image for decoding the next image. So the rotated image is written to other memory space. In this scheme, the display I/F has not to change memory space for displaying the decoded image because subsequent rotated image is written to the same space. The rotator modules support 8-types mode of 90 × n degree(n = 0, 1, 2, 3) rotating and mirroring.

Figure 61-13 gives architecture diagram of post rotation/mirroring module.

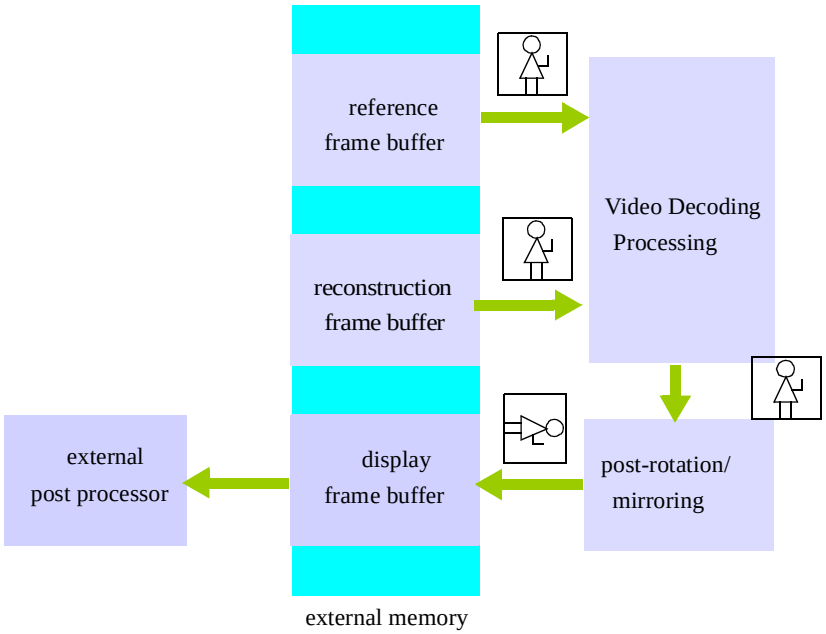


Figure 61-13. Post Rotation/Mirroring

61.4.2 Clocks

There are four clock domains in VPU, as follows.

- AXI bus clock domain (aclk)
  - Controls all AXI bus related functions
  - Maximum frequency is 166 MHz
  - Derives from the same clock as other modules performing AXI bus access in the i.MX51
  - Frequency determined by the whole system performance requirement.
- Decoding core clock domain (cclk)
  - VPU functional clock
  - Controls all VPU decoding functionality, with a maximum frequency of 166 MHz
  - Actual value of the core clock dependent upon maximum use case
  - Clock frequency scaling done through the VPU API
- IP bus clock domain (ipg_clk_s)
  - Controls VPU registers read/write function, with a maximum frequency of 66.5 MHz
  - Gated clock of IP green-line clock (ipg_clk) with ips_module_en
  - Turned off for power saving when there are no registers read/write
- Reference clock domain (rclk)—Used as reference clock for vpu_idle related logic.

VPU has all four clock domains because it involves in AXI signal generation, functional calculation, IP bus configuration, vpu_idle control logic. Only positive edge clocks are used in VPU design. Each clock is fully asynchronous with other clocks and separated from other clocks.

All clocks can be gated off when VPU is in the idle state to reduce power consumption. Aclk and cclk can be chosen by external system environment ad there is no restriction about the value of aclk and cclk. The actual frequency of aclk and cclk affects bus bandwidth and video decoding processing capability.

61.4.3 Reset

There are four reset signals going into VPU module, corresponding to four clock domains, active low.

- areset_n: used in AXI bus interface. Corresponding clock is aclk.
- creset_n: used in BIT processor and video decoder. Corresponding clock is cclk.
- ipg_hard_pos_async_reset_b: used in IP bus interface. Corresponding clock is ipg_clk_s.
- rreset_n: used in reference counter for vpu_idle or vpu_underrun. Corresponding clock is rclk.

The number of cycles for each reset signal must be at least 8 cycles.

VPU embeds an internal reset controller for feature of the software reset from the BIT processor. If any of the VPU blocks, with the exception of the BIT processor is needed to reset (software reset), the host processor can enable this software reset by setting the software reset register through VPU API. Besides, the BIT processor cannot be reset by this software reset scheme, because the reset signal of the BIT processor is connected directly to external reset signal.

If reset fires when VPU is processing a transaction through the AXI bus, there is no guarantee that the AHB will complete the transaction normally, since VPU will be reset. If there are any corrupted data in memory, it can be discarded by software. Basically, if the host processor needs to issue a reset, it must check that there is no transaction on AHB bus between the VPU and external AHB bus interface. In general, the AHB bus is free of VPU transactions when one frame decoding is completed. The start of next frame processing need software configuration.

61.4.4 Interrupts

There is one interrupt signal output from VPU. Basically, this interrupt is used to indicate completion of decoding one frame. It is generated when VPU interrupt is enabled and as well as the interrupt condition is met. The interrupt signal is active high and retains until the host processor clears it by writing “1” to the interrupt clear register. This signal is synchronized to the positive edge of aclk.

When getting frame completion interrupt, the software need to set the parameters for next frame processing and start the BIT processor again. The parameters are mainly the source/destination frame buffer base address. It can be different from the previous frame buffer since the previous completed frame of data may be needed for other image block like display block or IPU in the system.

Basically, the following operation responding to interrupt is dependent upon the application. For example, the software can send the decoded frame to the Image Processing Unit for post-processing and display, at the same time the software should prepare the next stream to be processed to the external memory before starting a new processing. This can be done through VPU driver.

61.4.5 Endianness

VPU supports both little- and big-endian memory systems. User should specify the endianness of the bitstream buffer and frame buffer corresponding to the application scenario. The endianness configuration can be done through VPU API.

61.5 Initialization Information

VPU embedded BIT processor is highly optimized to handle bitstream data. In addition to processing bitstreams, the BIT processor also controls the communication between VPU and host processor.

The BIT processor firmware to drive video decoding is divided into 2 parts.

One is boot code, which is downloaded by host processor through the IP bus to BIT processor internal memory, it’s loaded only once at the VPU initialization stage. The size of the boot code is 1 KB . This boot code has also to be written to a region of external memory, and the base address of the firmware region is written through VPU API.

The other is a package of firmware for driving decoding processing. Before starting decoding, firmware has to be written to the continuous region of external memory following the boot code. At run-time, the BIT processor will self download firmware into internal memory corresponding to the activated decoding standard, it’s loaded through the AXI bus.

The VPU initialization process is shown in Figure 61-14.

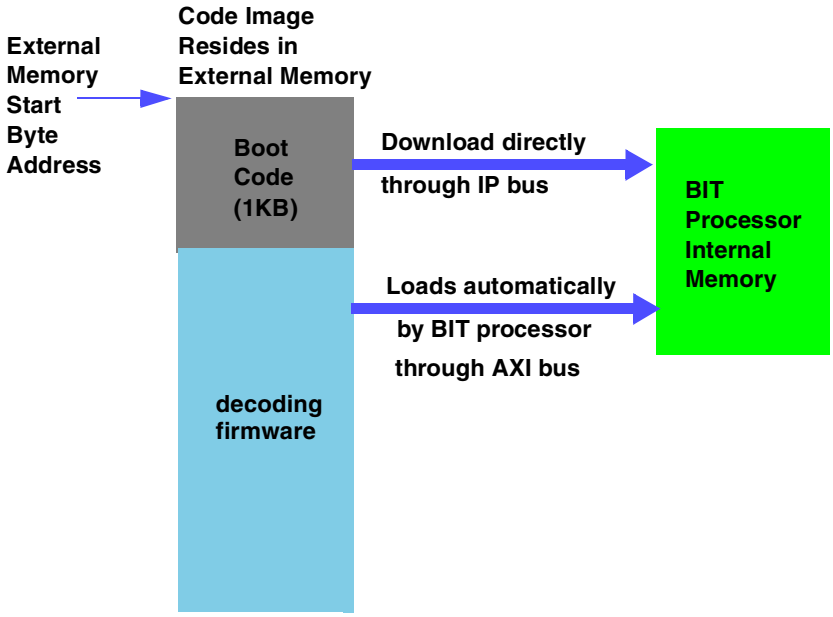


Figure 61-14. VPU Initialization Process

61.6 Application Information

Figure 61-15 shows roles of the BIT processor and VPU video processing core module and how to interface with application software. Basically, at the frame level, host processor communicates with VPU through the provided API's. To give the video decoding more flexibility and debugging capability, all processes related to the bitstream are assigned to the BIT processor.

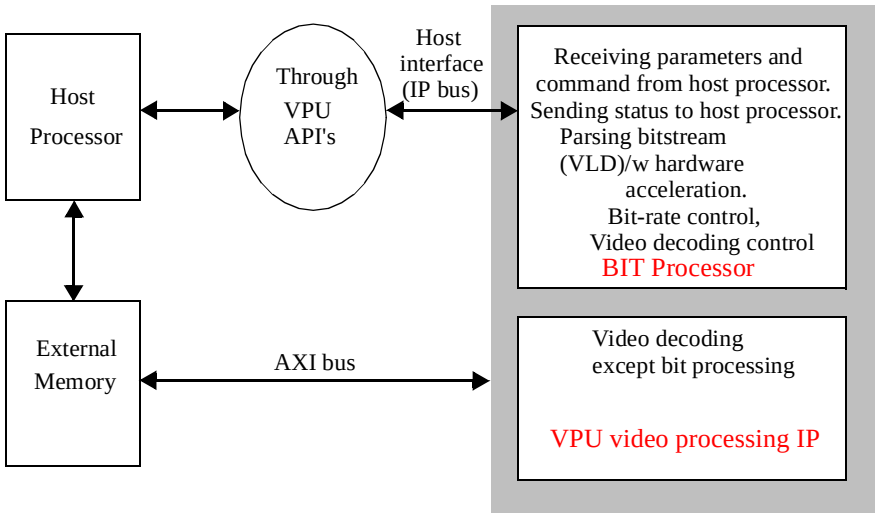


Figure 61-15. VPU interface with Application Software Diagram

61.6.1 Video Decoding Processing Control

This section describes how BIT processor controls the video decoding and communicates with the host.

61.6.1.1 Video Decoding Process Flow

VPU can handle a maximum of 4 processes simultaneously. Each process can have a different format - MPEG-4, MPEG-2, H.264 or VC-1. Figure 61-16 shows a simplified state diagram for running decoding process.

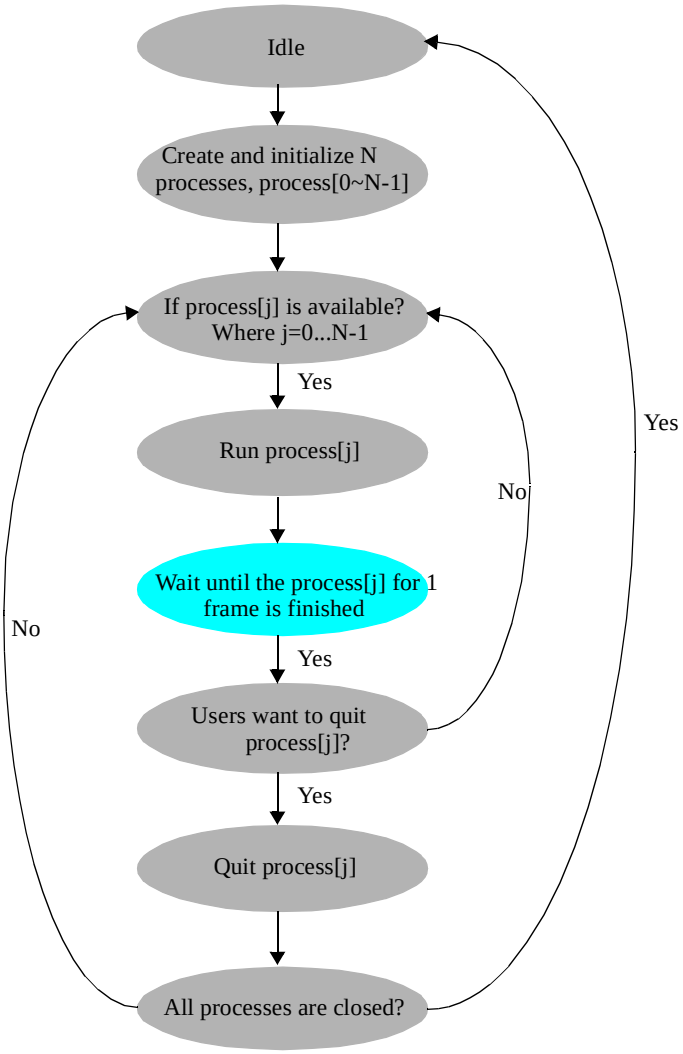


Figure 61-16. Decoding Process State Diagram

Each decoding process consists of the following three categories:

- Create processes—Software creates and configures processes.
- Running processes—At a proper time instance, software will begin a specific process. The proper time instance means when the decoding is in idle state and bitstream to be decoded is ready in the external memory.
- Quit Processes—Software can quit a specific process.

If more than one process is ready to run, each process must be assigned to different process ID - RunIndex, which is range from 0 to 3. Basically, the ID is assigned based on the order of creation. For example, when 1 MPEG-4 Decoding + 1 H.264 Decoding + 1 MPEG-2 Decoding + 1 VC1 Decoding are running simultaneously, MPEG-4 Decoding is assigned to process index “0”, H.264 Decoding is assigned to process index “1”, MPEG-2 Decoding is assigned to index “2”, and VC1 decoding is assigned to process index “3”.

There is no priority rules for executing processes, after creating all processes at the initialization step, the host enables the BIT processor to execute process specified with the RunIndex. All processes are executed in time-division like mechanism, after one process finishes decoding a frame, another process then can be executed.

In conjunction with the process ID, the RunCodStd needs to be set, to define which coding standard is used with the created process and whether the created process will decode a bitstream. Table 61-13 shows the dedicated RunCodStd value for each decoding standard. All this can be done through VPU API.

Table 61-13. RunCodStd Register Value For Coding Standard

Coding standard	RunCodStd
H264 decoding	0
VC-1 decoding	1
MPEG-2 decoding	2

61.6.1.2 Video Decoding Process Command

The following seven execution commands initialize, run, quit, set frame, and set parameter processes. A command is sent by writing the command value to the RunCommand register through VPU API.

- SEQ_INIT: This command is to initiate a decoding process. API should set following configuration parameters before sending this command to VPU.
  - Bitstream buffer base address and size
  - Frame buffers base addresses and stride lines
- SEQ_END: This command is to terminate a decoding process.
  - After this command, no more PICTURE_RUN commands will be accepted for this process.
- PICTURE_RUN: This command is to decode one picture.
  - In decoding case, frame destination address should be set before executing.
- SET_FRAME_BUF: This command informs frame buffer addresses, which are to be used as a decoding, to the BIT processor. Total 63 frame buffers may be used for decoding.

- The decoding image must be reserved for motion compensation reference until it is not used for reference. So the decoded frame buffer is re-used carefully. The BIT processor receives the whole frame buffer address by this command before picture decoding is started. Then the BIT processor manages the frame buffer allocation for next storage area for decoding.
- The frame buffer addresses are stored to external memory address. The luminance and two chrominance buffer addresses for each frame index must be stored.

- DEC_PARA_SET: This command adds a sequence parameter set or a picture parameter set to the decoder.
  - The sequence parameter set or the picture parameter set may be conveyed via “out-of-band”. In that case, the host must transfer the sequence parameter set or picture parameter set to decoder by this command.
  - The sequence parameter set or picture parameter set must be written to the Parameter buffer of the BIT processor in RBSP format prior to executing this command. The BIT processor decodes the transferred sequence/the picture parameter and stores decoded contents. The decoded sequence/picture parameter set will be activated at decoding slice header with the matched sequence/picture parameter set id.
  - Multiple sequence/picture parameter sets may be delivered to decoder. They are distinguished by different sequence/picture parameter set id. BIT processor can process 32 sequence parameter sets and 256 picture parameter sets.
  - The type (sequence or picture) and size (byte count) of conveyed sequence/picture parameter set must be delivered to BIT processor by command argument register.
- DEC BUF FLUSH: Flush data in bitstream buffer.
  - After this command finished, bitstream buffer read pointer will be 0. So host must set bitstream buffer write pointer as 0.
- GET F/W VERSION: Get firmware version.

There is a busy status register in the VPU to show whether the BIT processor is executing a command. The busy status keeps “1” until the command is finished, then the BIT processor can accept a new command.

61.6.1.3 Video Decoding Process Finish Detection

VPU raises the interrupt signal or busy state is asserted when a process finishes decoding a frame. So there are three ways of detecting whether the process is finished:

- Polling VPU interrupt status register. When interrupt is generated, the interrupt status register can indicate that.
- Polling VPU busy status register. When a PICTURE_RUN command is issued to BIT processor and the process is under operation, the busy status keeps “1”, as soon as the busy status becomes “0”, the decoding process is finished.
- Capture the interrupt signal from system level, respond to interrupt request within system interrupt service routine.

There is a single busy status register for all processes, user knows which process is finished by the specified running process ID—RunIndex. Interrupt status can be cleared by writing 1 to interrupt clear register. Busy state can be self-cleared after read out.

61.6.1.4 Video Decoding Process Flow Example

Figure 61-17 shows a process flow example for decoding an H.264 bitstream and decoding an MPEG4 bitstream simultaneously. At first both decoding process are created and initialized, then each process is executed with PICTURE_RUN command alternately. More details are described as below.

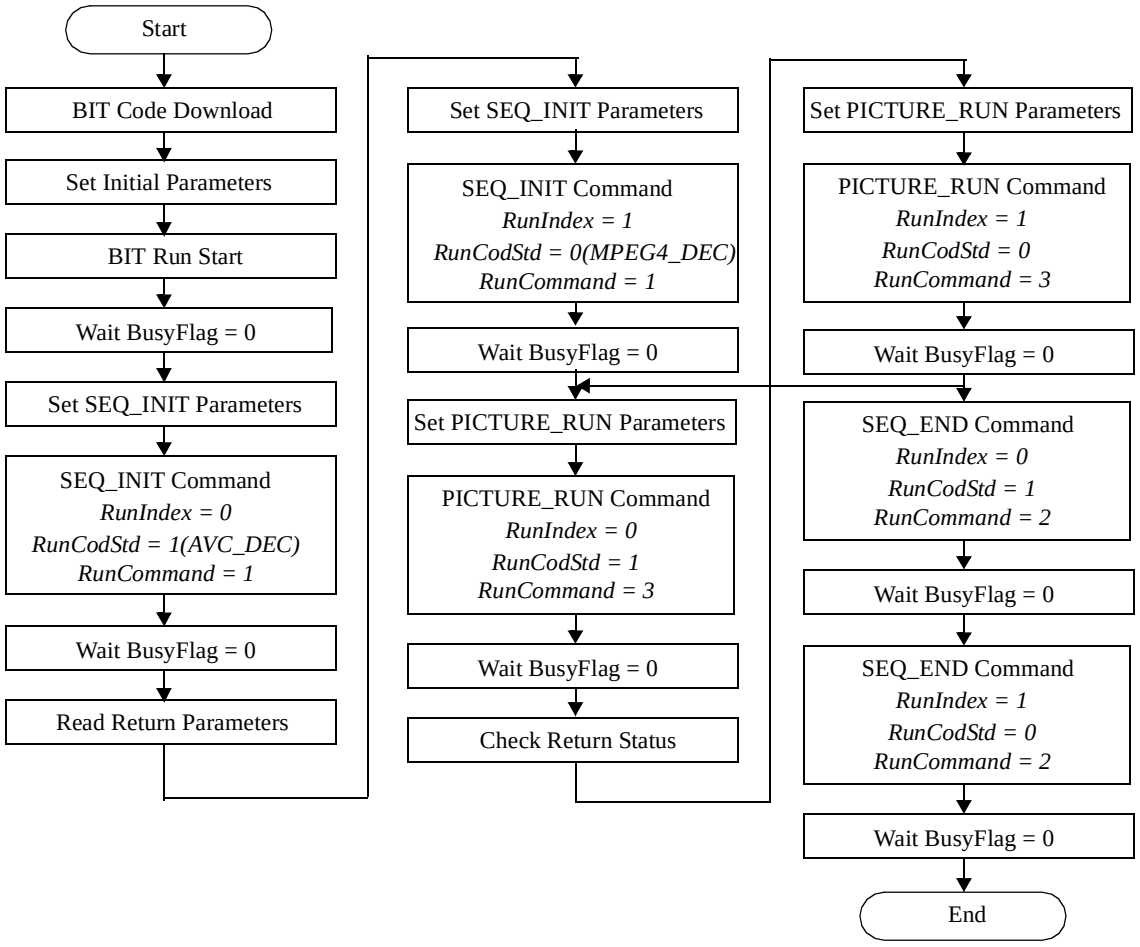


Figure 61-17. H.264 decoding and MPEG4 decoding process flow

*RunCommand = 1 (SEQ_INIT); RunCommand = 2 (SEQ_END);
RunCommand = 3 (PICTURE_RUN)

1. Initialize VPU
   - — BIT Code Download: Load BIT Processor firmware to memory.
   - — Set Initial Parameters: General configuration for BIT processor, setting working buffer base address, BIT Code memory address, bitstream buffer control and so on.
   - — BIT Run Start: Run BIT processor to initialize VPU.
2. Create and initialize an H.264 decoding process
   - — Set SEQ_INIT parameters: Configure base address and size of bitstream buffer, base address of frame buffers and so on.

- Run SEQ_INIT command: Initiate an H.264 decoding process.
- Wait BusyFlag=0: Wait BIT processor completes SEQ_INIT command execution.
- Read Return Parameters: Read the features of decoded bitstream, such as the picture resolution and number of reference frames, this can be done by reading the RetSrcFormat and Ret264Info register through VPU API. In this way, the host can prepare the required frame buffers.

3. Create and initialize an MPEG4 decoding process
   - The flow is similar to the H.264 decoding process except the run standard setting.
4. Run the H.264 decoding process
   - Set PICTURE_RUN Parameters: Configure the frame destination address.
   - Run PICTURE_RUN command: Start the H.264 decoding process.
   - Wait BusyFlag=0: Wait the BIT processor completes PICTURE_RUN command execution. It also means one frame process is finished. (The finish detection can be implemented in other way described in Section 61.6.1.3, Video Decoding Process Finish Detection) The decoded frame can be sent to the Image Processing Unit for post-processing and display. The actual operation is dependant on the application.
5. Run the MPEG4 decoding process
   - The flow is similar to the H.264 decoding process. The decoding process should configure frame source address in addition to destination address.
6. Execute step 4 and step 5 alternately.
   - Before running decoding process, the host should load new bitstream to the bitstream buffer if it is empty, and update the frame destination address according to the application.
7. Stop the decoding processes
   - Run SEQ_END command to each process to terminate it.

Basically, the process flow for decoding is similar, though it may have minor change for different firmware version. The detailed implementation is done in the VPU driver.

61.6.2 Video Encoding Processing Control

This section describes how BIT processor controls the video decoding and communicates with the host.

61.6.2.1 The Pipeline for Encoding

Figure 61-18 shows the six pipeline stages for H.264 encoding. In the encoding path, there are two separated data paths. The upper four stages are for the encoding path and the lower four stages are for reconstruction.

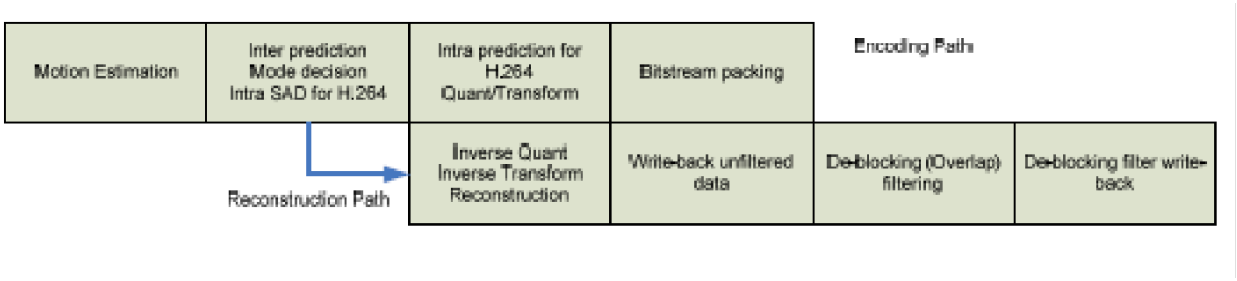


Figure 61-18. Encoding Pipeline Stage for H.264

The sub block in each stage is controlled by the BIT processor and the main controller (macroblock sequencer). The steps in the H.264 encoding pipeline stages are as follows.

- Pipeline stage 1
  a) The motion estimation (ME) block searches the motion vectors.
  b) The BIT processor reads the motion vectors from ME.
  c) The used data (reference luminance and current macroblock luminance data) is moved to the MC block for reuse.
- Pipeline stage 2
  a) The MC block writes the predicted data from ME motion vector.
  b) Intra predictor calculates the Intra SAD and cost for mode decision.
  c) Inter/intra prediction block writes the predicted data to predicted local memory for subtraction in the next pipeline stage.
- Pipeline stage 3
  a) Execute intra prediction and transfer the results to quantizer/transform.
  b) Quantized and transformed data is written to residual buffers for bitstream packing.
  c) Quantized and transformed data is transferred to inverse quantizer/inverse transform.
  d) Inverse quantized and inverse transformed data is added (reconstructed) with prediction block data to make reconstructed frame.
- Pipeline stage 4
  a) The BIT processor reads the residual buffer and makes the bitstream.
  b) Write back the unfiltered pixel data.
- Pipeline stage 5
  a) The deblocking filter reads the reconstructed data from its local memory and runs deblocking filtering for H.264/VC-1 or overlap smoothing for VC-1 intra block.

b) The filtered pixel data, that is the final reconstructed picture, is stored in the local memory ofb the deblocking filter block for write back.

- Pipeline stage 6
  a) The write-back DMA writes the decoded picture into external frame buffer.
  b) The decoded picture is now the reference picture for next picture decoding.

Figure 61-19 shows the five pipeline stages for MPEG-4/H.263 encoding when the deblocking filter is disabled and Figure 61-20 shows the six pipeline stages for H.263 when the deblocking filter is enabled. The encoding data path in MPEG-4/H.263 is similar to the H.264 encoding data path, with the exception of the inserting AC/DC prediction.

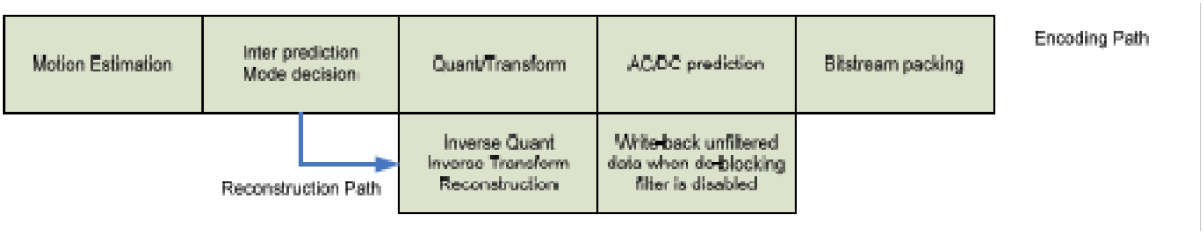


Figure 61-19. MPEG-4/H.263 Encoding Pipeline Stage when Deblocking Filter is Disabled

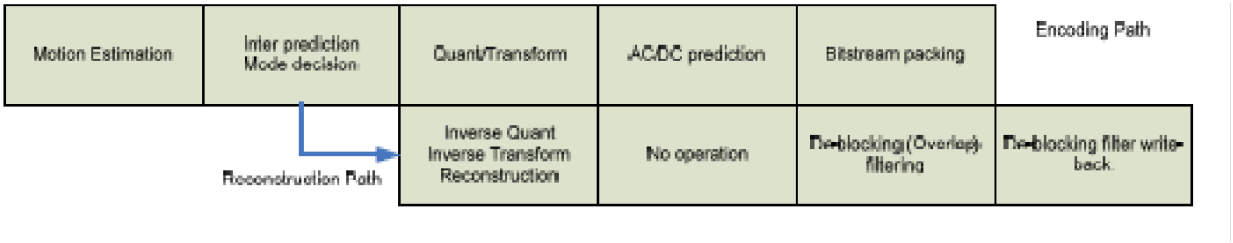


Figure 61-20. H.263 Encoding Pipeline Stage when Deblocking Filter is Enabled

61.6.3 Video Codec Processing Buffer Requirement

VPU has full access to the whole external memory. It uses external memory to load or store image frame, bitstream, program, and data for the BIT processor. AC/DC predication and deblocking filtering also use external memory. The buffer size requirement is dependent on the standard and target application. For example, the H.264 decoding uses multiple reference frames up to 16. MPEG-4 and H.263 decoding uses only one reference frame. Also each standard requires different temporary memory size when it processes deblocking or overlap-smoothing filtering.

VPU uses fthe ive kinds of buffer as follows:

- Frame Buffer—for storing image frame.
- BIT processor program memory—for boot code and firmware.
- Working Buffer—for intermediate data from the BIT processor and the video decoding hardware.
- Bitstream Buffer—for loading bitstream.

- Parameter Buffer—for BIT processor command execution argument and return data.
- Search RAM—for the use of ME module to reduce SDRAM bus loading.

Different buffers can be noncontiguous in external memory, although each buffer needs to be contiguous.

61.6.3.1 Frame Buffer

This section describes the memory map of the frame buffer and the size requirement.

As shown in Figure 61-21, a frame buffer is specified with the base address and the stride line. A complete image consists of Y, U, and V components. Therefore, an image needs three frame buffers for Y, U, and V components, these buffers can be noncontiguous in memory. The stride line means the width of the luminance component buffer in pixel unit must be a multiple of 8. The stride line for U and V frame buffers is a half of the Y frame buffer and is extracted automatically based on the stride line of the Y frame buffer. VPU supports 11-bits for stride line size.

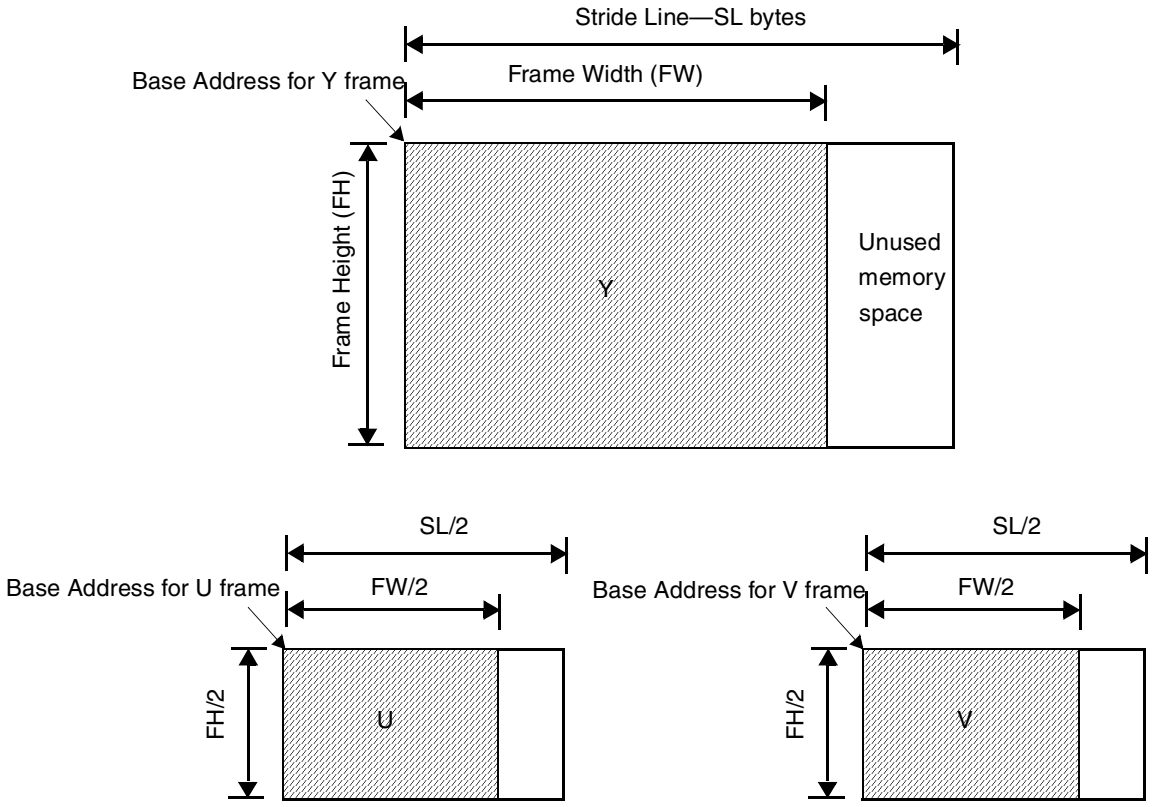
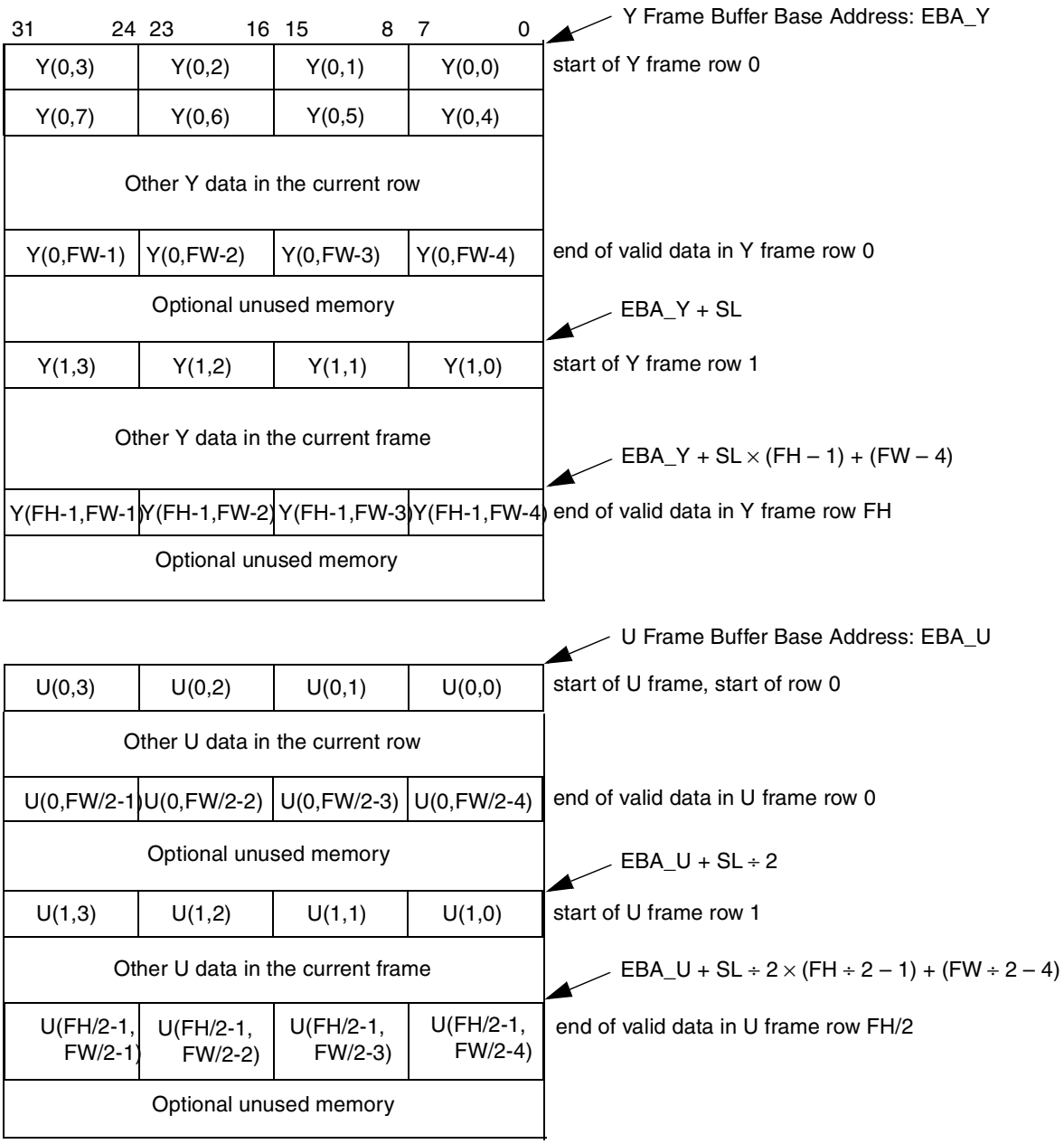


Figure 61-21. Frame Buffer configuration

Figure 61-22 shows the memory map of the frame buffer. For V frame buffer, the memory map is the same as the U frame buffer except the base address.

VPU supports both little and big endian system. It means Y(0,0) could be located in the bit[31:24]. User can specify the endianness through VPU API.



FH: Frame Height
SL: Stride Line
FW: Frame Width

Figure 61-22. Frame Buffer Address Map in Little Endian

Table 61-14 shows the frame buffer requirement for MPEG-4, H.264, and VC-1. The H.264 decoder requires multiple reference frames up to 16. In the case of VC-1 main profile, the decoder needs two reference frames to decode a B picture. Also, VC-1 requires two more frames that it stores for range reduction or multiresolution.

Table 61-14 also shows the memory requirement in case of QCIF/CIF/VGA resolution image. In the case of H.264 decoding, the required size for the reference frame is dependent on the level being supported. The VPU supports up to H264 decoding level 3.0, which the maximum decoded picture buffer size is defined as 3037.5 Kbytes in the standard. To support H.264 CIF at level 3.0, 2524 Kbytes is needed if 16 reference frames are used.

Table 61-14. Frame Buffer Requirement

		MEPG-4 Decoder	H.264 Decoder	VC-1 Decoder
QCIF	Reference Frames	2	16	2
	Instant Frames	1	1	2
	Display Frames	1	1	2
	Total Frames	4	18	6
	Picture Size[1]	37 Kbyte	37 Kbyte	37 Kbyte
	Total Frame Size	148 Kbyte	668 Kbyte	222 Kbyte
CIF	Reference Frames	2	16	2
	Instant Frames	1	1	2
	Display Frames	1	1	2
	Total Frames	4	18	6
	Picture Size	148 Kbyte	148 Kbyte	148 Kbyte
	Total Frame Size	592 Kbyte	2672 Kbyte	891 Kbyte
VGA	Reference Frames	2	5	2
	Instant Frames	1	1	2
	Display Frames	1	1	2
	Total Frames	4	7	6
	Picture Size	450 Kbyte	450 Kbyte	450 Kbyte
	Total Frame Size	1800 Kbyte	3150 Kbyte	2700 Kbyte

1 The picture size is the minimum size of one frame buffer with the assumption that the picture is YUV 4:2:0 format and the stride line is equal to frame width.

61.6.3.2 BIT Processor Program Buffer

At the initialization stage of VPU, the host processor must download boot code to the BIT processor. After initialization, the BIT processor loads a program corresponding to the activated standard. The program size of the boot code is 1 Kbyte. The current version of BIT firmware has a total 48 Kbyte size to support multistandards decoding.

61.6.3.3 Working Buffer

Besides buffers for frames and firmware, additional working buffer for intermediate data from the BIT processor and decoding is needed. The buffers are such as the reconstructed pixel row buffer for MPEG-4 AC/DC prediction or H.264 intra prediction, context saving buffer for running multiple processes and various temporal storage buffer for decoding process.

The required working buffer size varies according to decode size, decoding standard, decoding capability. For example, AC/DC prediction buffer size is determined by picture width and the maximum bitstream re-ordering buffer for data partition is determined by the maximum bitstream size of one picture. The working buffer size may change for different firmware version. The current version of firmware requires a maximum 256 Kbyte for working. Its size can be set through the VPU API. The detailed working buffer is organized as Table 61-15.

Table 61-15. Working Buffer Organization

Working Buffer Type	Name	Description	Size
Static buffer	STATIC_PRC_DMEM	BIT processor data memory of each process for context switching.	5 Kbytes for each process
	STATIC_PRC_SEQ	Static data storage of each sequence.	8 Kbytes for each process
Temp avc decoding buffer	AVC_DEC_PRED	Neighboring prediction buffer for bit stream parsing	204 Kbytes
	AVC_DEC_IP	Intra prediction buffer of Y/Cb/Cr	
	AVC_DEC_DBK	Deblocking filter buffer	
	AVC_DEC_SLICE	Slice data RBSP buffer. All slice data RBSP of one picture is stored.	

61.6.3.4 Bitstream Buffer

The host processor has to assign buffers for bitstreams on a per instance basis. If VPU handles N-bitstreams simultaneously in an application, the host should assign N bitstream buffers and specify the base address and size. The External bitstream buffer is “ring buffer” type. The start address of ring buffer and buffer size must be written by host to BIT processor. The current read or write address of ring buffer is automatically wrapped-around by firmware.

In decoding case, the host writes the bitstream to be decoded then BIT processor reads bitstream. In this case, the bitstream overwriting or underflow may occur and if it occurs, decoding will fail. To prevent overwriting or underflow, current bitstream read/write pointer must be exchanged between the host and the BIT processor.

The BIT processor writes current read pointer of ring buffer to internal register, and the host must write current write pointer of ring buffer to internal register. The BIT processor checks the bit buffer empty (underflow) status by comparing current read pointer and write pointer. If no more bitstream data is available to be decoded (buffer empty status), the BIT processor stops bitstream decoding to prevent misreading the bitstream and waits until the host writes more bitstream data and updates write pointer. The

host must check the current read pointer and write pointer before writing more bitstream data to ring buffer to prevent overwriting bitstream data.

61.6.3.5 Parameter Buffer

Host processor must reserve parameter buffer in external memory for BIT processor command execution argument and return data.

61.6.3.6 Search RAM

The CodaHx14's motion estimation module uses a search RAM to reduce the bandwidth on the external SDRAM. Generally, motion estimation reads a reference pixel data several times. To avoid this bandwidth overhead, the motion estimation module loads the reference pixel data form the external SDRAM one time and stores them to the search RAM through AXI bus. The stored reference pixel data is loaded several times to search the motion vector, but these operations are conducted through AMBA AXI bus. Therefore the bandwidth of loading reference pixel data is reduced by using search RAM in AXI bus.

The size of search RAM is as follows:

(# of horizontal pixel × 36 + 8 macroblocks) bytes.

For example ,in CIF encoding, the search RAM size is 352 × 36 + 2048 = 14720 bytes = 14.375 Kbyte.

61.6.3.7 Buffer Requirement Summary

Table 61-16 shows a summary of buffer requirement for each decoding instance, where bitstream buffer size is not considered because the size is not limited. The total size may change for different firmware version. Except the BIT processor program memory, other kinds of buffer has to be assigned on a per instance basis. The overall buffer size for a multiparty call application is nearly the sum of each instanced decoding.

Table 61-16. Summary of Buffer Requirement

		MEPG-4 Decoder	H.264 Decoder	VC-1 Decoder
QCIF	Frames buffer	148 Kbytes	668 Kbytes	222 Kbytes
	BIT processor program buffer	128 Kbytes		
	Parameter buffer	8 Kbytes		
	Bitstream buffer	1000 Kbytes		
	Working Buffer	13 Kbytes	256 Kbytes	13 Kbytes
	Total	1297 Kbytes	2060 Kbytes	1371 Kbytes

Table 61-16. Summary of Buffer Requirement (continued)

		MEPG-4 Decoder	H.264 Decoder	VC-1 Decoder
CIF	Frames Size	592 Kbytes	2672 Kbytes	891 Kbytes
	BIT processor program buffer	128 Kbytes		
	Parameter buffer	8 Kbytes		
	Bitstream buffer	1000 Kbytes		
	Working Buffer	13 Kbytes	256 Kbytes	13 Kbytes
	Total	1741 Kbytes	4064 Kbytes	2040 Kbytes
VGA	Frame Size	1800 Kbytes	3150 Kbytes	2700 Kbytes
	BIT processor program buffer	128 Kbytes		
	Parameter buffer	8 Kbytes		
	Bitstream buffer	1000 Kbytes		
	Working Buffer	13 Kbytes	256 Kbytes	13 Kbytes
	Total	2949 Kbytes	4542 Kbytes	3849 Kbytes

Chapter 62
Watchdog Timer (WDOG)

The watchdog (WDOG) timer module protects against system failures by providing a method of escaping from unexpected events or programming errors.

62.1 Overview

Once the WDOG module is activated, it must be serviced by the software on a periodic basis. If servicing does not take place, the timer times out. There is also an option for programmable interrupt generation before the counter actually times out. The time at which the interrupt needs to be generated prior to counter timeout can be programmed through watchdog interrupt control register (WICR).

Upon a time-out, the WDOG timer module asserts the internal system reset signal, wdog_rst, which will be going to CCMs. This signal can also be asserted via a software write to the watchdog control register (WCR). There is a power-down counter that is enabled out of any reset (POR, warm/cold). This counter has a fixed time-out period of 16 seconds after which it asserts the ipp_wdog signal. The ipp_wdog is also asserted by time-out counter expiration if WDT bit in WCR (WDOG control register) is set. Flow diagrams for the time-out counter power down counter and interrupt operations are shown in Figure 62-10, Figure 62-11, and Figure 62-12.

The input clock ipg_async_ckil_clk is the asynchronous 32 kHz clock. The power down counter works on this clock.

NOTE

ipg_clk_32k is the synchronized version of the Async. CKIL on the IP global functional clock. Because the synchronized version is not available (the IP global functional clock is off) in low-power mode, it will receive the raw CKIL clock from the CCM. These synchronizers will be bypassed while going into low power modes in CCM.The WDOG module can continue or suspend the timer operation in the low power modes.

Figure 62-1 shows the WDOG block diagram.

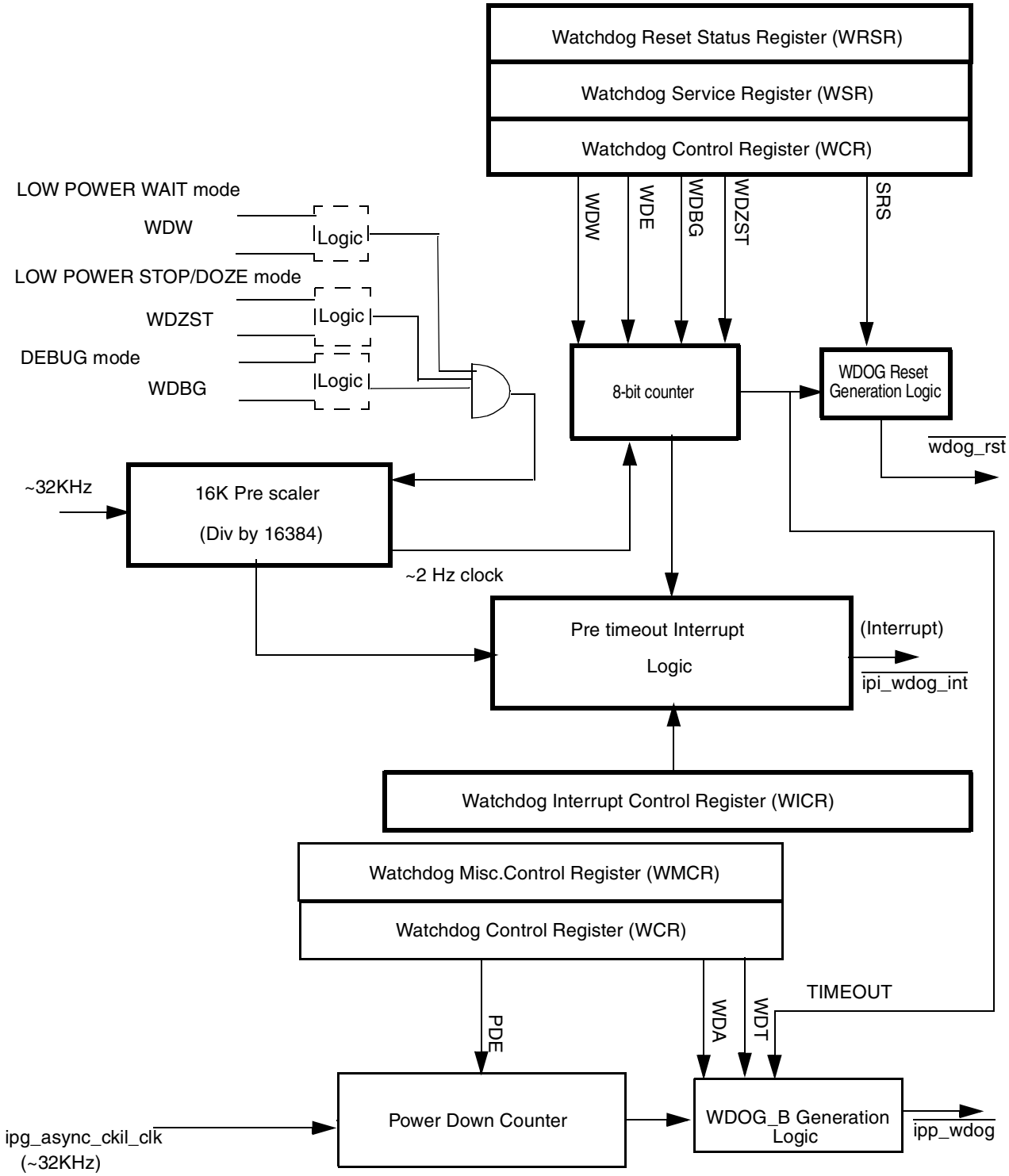


Figure 62-1. WDOG Block Diagram

62.2 Features

The WDOG features are as follows:

- A time-out counter with time-out periods from 0.5 seconds up to 128 seconds
- Time resolution of 0.5 seconds
- Configurable time-out counter that can be programmed to run or stop during low-power modes
- Configurable time-out counter that can be programmed to run or stop during DEBUG mode
- Programmable interrupt generation prior to time-out.
- Programmable time duration between interrupt and timeout events, from 0 to 128 seconds in steps of 0.5 seconds.
- A power down counter with fixed time-out period of 16 seconds, which if not disabled after reset deassertion asserts $\overline{\text{ipp\_wdog}}$ signal low
- Power down counter enabled out of any reset (POR, warm/cold reset) by default

62.3 External Signal Description

The WDOG module port signals going to pins are listed in Table 62-1.

Table 62-1. Signal Properties

Name	Port	Function	Reset State	Pull up
$\overline{\text{ipp\_wdog}}$	—	Asserted by software, power down counter timeout and timeout condition.	1	—

The $\overline{\text{ipp\_wdog}}$ signal can be routed to external pin of the IC. It is asserted by a software request (setting the WCR bits), by power down counter timeout, and timeout counter expiration.

62.4 Memory Map and Register Definitions

The WDOG module has user-accessible, 16-bit registers used to configure, operate, and monitor the state of the Watchdog Timer. Section 62.5, Register Descriptions," provides the detailed descriptions for all of the WDOG registers.

62.4.1 Watchdog Timer Memory Map

Table 62-2 shows the WDOG memory map. Watchdog uses 16-bit registers. All registers are byte writable except WRSR, which is a read only register.

Table 62-2. WDOG Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x000 (WCR)	Watchdog Control Register	R/W	0x0030	62.5.1/62-5

Table 62-2. WDOG Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x002 (WSR)	Watchdog Service Register	R/W	0x0000	62.5.2/62-7
0xBASE+0x004 (WRSR)	Watchdog Reset Status Register	R	0x00XX	62.6.5/62-12
0xBASE+0x006 (WICR)	Watchdog Interrupt Control Register	R/W	0x0004	62.5.4/62-8
0xBASE+0x008 (WMCR)	Watchdog Miscellaneous Control Register	R/W	0x0001	62.5.4/62-8

62.4.2 Register Summary

Figure 62-2 shows the key to the register fields and Table 62-3 shows the register figure conventions.

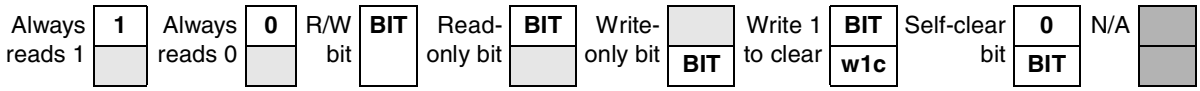


Figure 62-2. Key to Register Fields

Table 62-3. Register Figure Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
r	Read only. Writing this bit has no effect.
w	Write only.
rw	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero.
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 62-4 shows the WDOG register summary.

Table 62-4. WDOG Register Summary

Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x000 (WCR)	R	WT								WDW	0	WDA	SRS	WDT	WDE	WDBG	WDZST
	W																
0xBASE+0x002 (WSR)	R	WSR															
	W																
0xBASE+0x004 (WRSR)	R	0	0	0	0	0	0	0	0	0	0	?	?	?	?	TOUT	SFTW
	W																
0xBASE+0x006 (WICR)	R	WIE	WTIS	0	0	0	0	0	0	WICT[7:0]							
	W		w1c														
0xBASE+0x008 (WMCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PDE
	W																

62.5 Register Descriptions

This section provides detailed descriptions of the watchdog registers.

62.5.1 Watchdog Control Register (WCR)

The Watchdog Control Register (WCR) is a 16-bit read/write register. It controls the WDOG operation. Figure 62-3 shows the register; Table 62-5 provides its field descriptions.

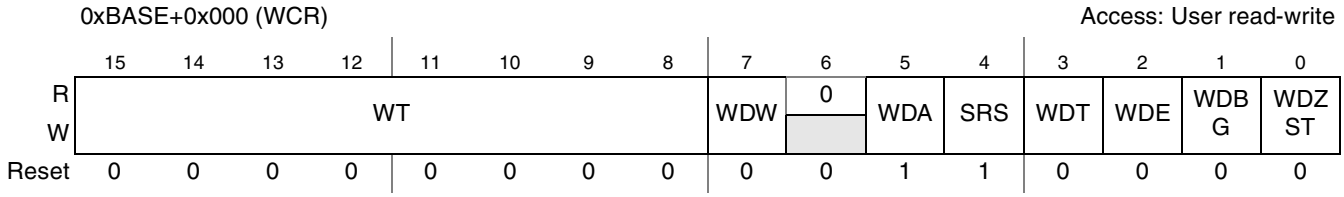


Figure 62-3. Watchdog Control Register

NOTE

WDZST, WDBG and WDW bits are write-once only bits. Once the software does a write access to the WCR Register, these bits are locked and cannot be reprogrammed. WDE and WDT are write one once only bits. Once software sets them, they cannot be reset.

WDT bit gets reset only on POR(Power on Reset).

Table 62-5. WCR Register Descriptions

Field	Description
15–8 WT	Watchdog Timeout Field. This 8-bit field contains the time-out value that is loaded into the Watchdog counter after the service routine has been performed or after the Watchdog is enabled. After reset, WT[7:0] must have a value written to it before enabling the Watchdog otherwise count value of zero which is 0.5 seconds is loaded into the counter and the new timeout value takes effect after it has been serviced. 0x00 - 0.5 Seconds(Default). 0x01 - 1.0 Seconds. 0x02 - 2.0 Seconds. 0x03 - 2.5 Seconds. ..... 0xff - 128 Seconds. **NOTE:** The new timeout value is reloaded to the counter after the WDOG is enabled or after the service routine to WSR(WDOG service register) has been performed.
7 WDW	Watchdog Disable for Wait.This bit determines the operation of WDOG during Low Power WAIT mode. This is a write once only bit. 0 Continue WDOG timer operation(Default). 1 Suspend WDOG timer operation.
6	Reserved
5 WDA	$\overline{\text{ipp\_wdog}}$ assertion. Controls the software assertion of the $\overline{\text{ipp\_wdog}}$ signal. 0 Assert $\overline{\text{ipp\_wdog}}$ output. 1 No effect on system(Default).
4 SRS	Software Reset Signal. Controls the software assertion of the WDOG-generated reset signal. This bit automatically resets to “1” after it has been asserted to “0”. 0 Assert system reset signal. 1 No effect on the system(Default).
3 WDT	$\overline{\text{ipp\_wdog}}$ Timeout assertion. Determines if the $\overline{\text{ipp\_wdog}}$ gets asserted upon a Watchdog Timeout Event. 0 No effect on ipp_wdog(Default). 1 Assert $\overline{\text{ipp\_wdog}}$ upon a Watchdog Timeout event. **Note:** There is no effect on $\overline{\text{wdog\_rst}}$ upon writing on this bit. $\overline{\text{ipp\_wdog}}$ gets asserted along with $\overline{\text{wdog\_rst}}$ if this bit is set.
2 WDE	Watchdog Enable. Enables or disables the WDOG module. Software can only write “1” in this bit. It is not possible to reset this bit by a software write, once the bit is set. **Note:** This bit can be set/reset in debug mode (exception). 0 Disable the Watchdog. 1 Enable the Watchdog.
1 WDBG	Watchdog DEBUG Enable. Determines the operation of the WDOG module during DEBUG mode. This bit is write once-only. 0 Continue WDOG timer operation(Default). 1 Suspend the watchdog timer.
0 WDZST	Watchdog Low Power. Determines the operation of the WDOG module during low-power modes. This bit is write once-only. **Note:** The WDOG module can continue/suspend the timer operation in the low-power modes (STOP mode). 0 Continue timer operation(Default). 1 Suspend the watchdog timer.

62.5.2 Watchdog Service Register (WSR)

When enabled, the WDOG requires that a service sequence be written to the Watchdog Service Register (WSR). Figure 62-4 shows the register; Table 62-6 provides its field descriptions.

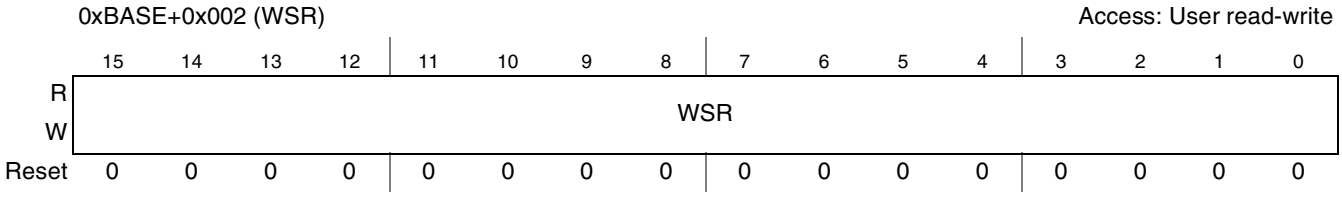


Figure 62-4. Watchdog Service Register (WSR)

Table 62-6. Watchdog Service Register Description

Field	Description
15–0 WSR	Watchdog Service Register. This 16-bit field contains the Watchdog service sequence. Both writes must occur in the order listed prior to the time-out, but any number of instructions can be executed between the two writes. The service sequence must be performed as follows: Write 0x 5555 to the Watchdog Service Register (WSR) Write 0x AAAA to the Watchdog Service Register (WSR)

NOTE

Executing the service sequence will reload the WDOG time out counter irrespective of whether interrupt has been generated or not.

62.5.3 Watchdog Reset Status Register (WRSR)

The WRSR is a read-only register that records the source of the output reset assertion. It is not cleared by a hard reset. It records the source of the output reset assertion. Therefore, only one bit in the WRSR will always be asserted high. The register will always indicate the source of the last reset generated due to WDOG. Any write performed on this register will generate an ips_xfr_err.

A reset can be generated by the following sources, as listed in priority from highest to lowest:

- Watchdog Timeout
- Software Reset

:

Figure 62-5 shows the register; Table 62-7 provides its field descriptions.

0xBASE+0x004 (WRSR) Access: User read-only

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	TOUT	SFTW
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	—	—

Figure 62-5. Watchdog Reset Status Register (WRSR)

Table 62-7. Watchdog Reset Status Register Description

Field	Description
15–2	Reserved
1 TOUT	Time-out. Indicates whether the reset is the result of a WDOG time-out. 0 Reset is not the result of a WDOG time-out. 1 Reset is the result of a WDOG time-out.
0 SFTW	Software Reset. Indicates whether the reset is the result of a WDOG software reset by asserting SRS bit 0 Reset is not the result of a software reset. 1 Reset is the result of a software reset.

62.5.4 Watchdog Interrupt Control Register (WICR)

The WICR is a 16-bit read/write register. It controls the WDOG interrupt generation. The reset value for WIE bit in WICR is 0 and that of WICT field is 0x04. WTIS bit will reflect the status whether the interrupt has occurred or not.

Figure 62-6 shows the register. Table 62-8 provides its field descriptions.

0xBASE+0x006 (WICR) Access: User read-write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WIE	WTIS	0	0	0	0	0	0	WICT[7:0]							
W		w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Figure 62-6. Watchdog Interrupt Control Register (WICR)

Table 62-8. Watchdog Interrupt Control Register Description

Field	Description
15 WIE	Watchdog Timer Interrupt enable bit. Reset value is 0. This bit is write once only 0 Disable Interrupt(Default). 1 Enable Interrupt.
14 WTIS	Watchdog TImer Interrupt Status bit will reflect the timer interrupt status, whether interrupt has occurred or not.Once the interrupt has been triggered software must clear this bit by writing 1 to it. 0 No interrupt has occurred(Default). 1 Interrupt has occurred

Field	Description
13–8	Reserved
7–0 WICT	Watchdog Interrupt Count Timeout (WICT) field determines, how long before the counter timeout must the interrupt occur. The reset value is 0x04 implies interrupt will occur 2 seconds before timeout. The maximum value that can be programmed to WICT field is 127.5 seconds with a resolution of 0.5 seconds.This field is write-once only. WICT[7:0] = 0x00 Time duration between interrupt and timeout is 0 seconds. WICT[7:0] = 0x01 Time duration between interrupt and timeout is 0.5 seconds. ... WICT[7:0] = 0x04 Time duration between interrupt and timeout is 2 seconds(Default). ... WICT[7:0] = 0xff Time duration between interrupt and timeout is 127.5 seconds.

62.5.5 Watchdog Miscellaneous Control Register (WMCR)

Figure 62-7 shows the Register. Table 62-9 shows the description.

	0xBASE+0x008 (WMCR)												Access: User read-write			
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Figure 62-7. Watchdog Miscellaneous Control Register (WMCR)

Table 62-9. Watchdog Miscellaneous Control Register Description

Field	Description
15–1	Reserved
0 PDE	Power Down Enable bit. Reset value of this bit is 1 means the power down counter inside WDOG is enabled. The software must write 0 to this bit to disable the counter within 16 seconds of reset deassertion. Once disabled this counter cannot be enabled again. 0 Power Down Counter of WDOG is disabled. 1 Power Down Counter of WDOG is enabled.(Default)

62.6 Functional Description

This section describes the timing information for the WDOG module.

62.6.1 Timing Specifications

62.6.1.1 Time-out Event

The WDOG provides time-out periods from 0.5 seconds up to 128 seconds with a time resolution of 0.5 seconds. It uses the ipg_clk_32k clock (32.768 kHz frequency clock) as an input to prescalers. The prescalers divide the clock by a fixed value of 16384 (div by 16K) to achieve the resolution of 0.5 seconds

and a frequency of 2 Hz. The output of the prescaler circuitry is connected to the input of a 8-bit counter to obtain a range of 0.5 to 128 seconds. The user can determine the time-out period by writing to the WDOG Time-out field (WT[7:0]) in the WDOG Control Register (WCR). The WDOG has to be enabled for the timeout counter to start running. After the timeout, the WDOG outputs a system reset signal and asserts ipp_wdog if WDT bit in WCR(WDOG Control register) is set. Detailed description of timeout condition and its prevention can be found in the Section 62.6.3, Watchdog After Reset."

62.6.1.2 Interrupt Event

Prior to time-out the WDOG can generate an interrupt which can be considered as a warning signal to indicate that the time-out will occur shortly. The interrupt will be generated only if the WIE bit in WICR Register is set. The interrupt will be deasserted only if software clears the WTIS bit in WICR Register. The time duration between interrupt event and time-out event can be controlled by writing to the WICT field of WICR Register. It can vary between 0 and 128 seconds. If the WDOG is serviced (Section 62.6.3.3, Servicing the WDOG to Reload the Counter") before the interrupt generation then the counter will be reloaded with the timeout value WT[7:0] of WCR and interrupt won't be triggered.

62.6.1.3 Power-down Counter Event

Soon after reset, the power down counter in WDOG starts running and expires after 16 seconds thereby asserting ipp_wdog. To prevent this, the software must disable this counter by clearing the PDE bit of WMCR register within 16 seconds of reset deassertion. Once disabled this counter cannot be enabled again until the next system reset occurs. This feature is provided to prevent hanging up of cores after reset.

62.6.2 Watchdog During Reset

A system reset (cold/warm) resets all registers (except the WRSR and WDT bit in WCR) to their initial default values, and places the time-out counter in the idle state until the WDOG is enabled. The Watchdog Reset Status Register (WRSR) contains the source of the reset events which are triggered from WDOG The WDT bit in WCR gets reset only on POR (Power on Reset) and not by a system reset.

62.6.3 Watchdog After Reset

The following subsections define the WDOG Time-out Counter state after reset.

62.6.3.1 Initial Load

The initial load into the timeout counter is done by setting the Watchdog Enable (WDE) bit in the WCR. The WT[7:0] field of Watchdog Control Register (WCR) must have a time-out value written to it before the watchdog is enabled. Otherwise the count value of zero timeout value, which is 0.5 seconds, is loaded into the counter, and the new timeout value takes effect after it has been serviced.

Another way of loading the time-out value into the counter is the service sequence being written to the Watchdog Service Register (WSR). The service sequence is described in the Section 62.6.3.3, Servicing the WDOG to Reload the Counter. The counter flow diagram is shown in Figure 62-10.

62.6.3.2 Timer Countdown

After the WDOG is enabled, the counter is activated and begins to count down from its initial programmed value. The timer will time-out when the counter reaches zero. The WDOG outputs a system reset signal and asserts ipp_wdog (WDT bit should be set in WCR register) after the counter reaches zero.

However the Timeout condition can be prevented by reloading the counter with the new timeout value(WT[7:0] of WCR) if a service routine (See Section 62.6.3.3, Servicing the WDOG to Reload the Counter) is performed before the counter reaches zero. If any system errors occur that prevent the software from servicing the Watchdog Service Register (WSR), then the timeout condition occurs.

By performing the service routine, the WDOG reloads its counter to the time-out value indicated by bits WT[7:0] of the WCR, and it re-starts the countdown. A system reset will reset the counter and place it in the idle state at any time during the countdown. The counter flow diagram is shown in Figure 62-10.

62.6.3.3 Servicing the WDOG to Reload the Counter

The proper service sequence to reload a time-out value to the counter begins by writing 0x5555 followed by 0xAAAA to the WSR. To reload the counter, the IP writes must take place within the time-out value indicated by bits WT[7:0] of the WCR. Any number of instructions can be executed between the two writes. This service sequence is also used to activate the counter during the initial load. See Section 62.6.3.1, Initial Load.” The time-out value can be changed at any point of the time and the counter reloads the new timeout value whenever the core services the Watchdog.

If the WSR is not loaded with 0x5555 prior to writing 0xAAAA to the WSR, the counter is not reloaded. If any value other than 0xAAAA is written to the WSR after 0x5555, the counter is not reloaded. If WDOG is serviced before the interrupt generation event, then both time out and interrupt are prevented from triggering. This service sequence reloads the counter with the timeout value WT[7:0] of WCR at any point of time.

62.6.3.4 Interrupt

WDOG generates an interrupt before timing out if the WIE bit is set in WICR. The exact time at which the interrupt should occur prior to timeout depends on the value of WICT field of WICR register. For example, if the WICT field has a value 0x04, the interrupt will be generated 2 seconds prior to time-out. If the WDOG is serviced before the interrupt generation, the counter will be reloaded with the timeout value WT[7:0] of WCR, and an interrupt will not be triggered. Once the interrupt is triggered, the WTIS bit in WICR Register will be set. The software needs to clear this bit so as to service the interrupt.

62.6.3.5 Power-down Counter after Reset

The power-down counter inside WDOG is enabled out of reset. This counter has a fixed time out value of 16 seconds, after which it drives the ipp_wdog signal low. To prevent this PDE bit in WMCR must be cleared by the software.

62.6.4 Low-Power and DEBUG Modes

The WDOG module can either continue or suspend the timer operation during low-power modes (WAIT and STOP) and DEBUG mode.

62.6.4.1 Low-Power Mode (WAIT, STOP)

The WDOG timer can be configured for continual operation or suspended in low-power WAIT/STOP mode.

62.6.4.1.1 STOP Mode

If the WDOG Timer Disable bit for low power STOP mode (WDZST) bit in the WCR is 0, the WDOG Timer continues to operate using the ipg_clk_32k clock (32.768 kHz frequency clock). If the low-power STOP Enable (WDZST) bit is set to 1, the WDOG operation is suspended. Upon exiting low-power STOP mode, the WDOG operation returns to what it was prior to entering the STOP mode.

62.6.4.1.2 WAIT Mode

If the WDOG Timer Disable bit for the low-power WAIT mode (WDW) bit in the WCR is 0, the WDOG Timer continues to operate using the ipg_clk_32k clock (32.768 kHz frequency clock). If the low-power WAIT Enable (WDW) bit is set to 1, the WDOG operation is suspended. Upon exiting low-power WAIT mode, the WDOG operation returns to what it was prior to entering the WAIT mode.

62.6.4.2 DEBUG Mode

The WDOG Timer can be configured for continual operation, or the operation can be suspended during debug mode. If the WDOG debug enable (WDBG) bit is set to 1 in the Watchdog Control Register (WCR), the WDOG module operation is suspended in debug mode. After 2 CKIL clocks of ipg_debug signal assertion, the counter is stopped, but register read and write accesses continue to function normally. Also, while in DEBUG mode, the WDE bit can be enabled-disabled directly.

NOTE

If the WDE bit is cleared while in DEBUG mode, it remains cleared even after exiting DEBUG mode. If the WDE bit is not cleared while in DEBUG mode, the WDOG count continues from its value before DEBUG mode was entered.

62.6.5 Watchdog Reset Control

The WDOG generated reset signal $\overline{\text{wdog\_rst}}$ can be asserted by the following:

- A software write to the Software Reset Signal (SRS) bit of the WCR.
- WDOG time-out. See Section 62.6.1.1, Time-out Event.

The $\overline{\text{wdog\_rst}}$ is generated for 1 clock cycle of ipg_clk_32k for a time-out. In case of a software reset, the $\overline{\text{wdog\_rst}}$ is asserted for 1 clock cycle of ipg_clk_32k (IP global functional clock). It remains asserted for for 1 clock cycle of ipg_clk_32k even if a system reset is asserted in between.

The watchdog-generated reset signal wdog_rst is an output to the for system reset generation.

NOTE

The of the IC generates the system reset signal on assertion of wdog_rst.

62.6.6 ipp_wdog Operation

The WDOG module asserts ipp_wdog a under the following scenarios.

- Software write to WDA bit of WCR.
- WDOG timeout condition, WDT bit in WCR(WDOG Control register) must be set for this scenario. Description of timeout condition can be found in the Section 62.6.1.1, Time-out Event."
- WDOG Power down Counter time-out, PDE bit of WMCR should not be cleared for this scenario. Description of this counter can be found in the Section 62.6.1.3, Power-down Counter Event."

If asserted by a software write to the WDA bit, it remains asserted as long as the WDA bit is "0". The power down counter timeout asserts it low until next system reset. Whenever a ipp_wdog is asserted due to a time-out condition, it remains asserted until a POR (Power-on reset) occurs. It is cleared after the POR occurs and not due to any other system reset.

Figure 62-8 shows the scenarios under which ipp_wdog is asserted

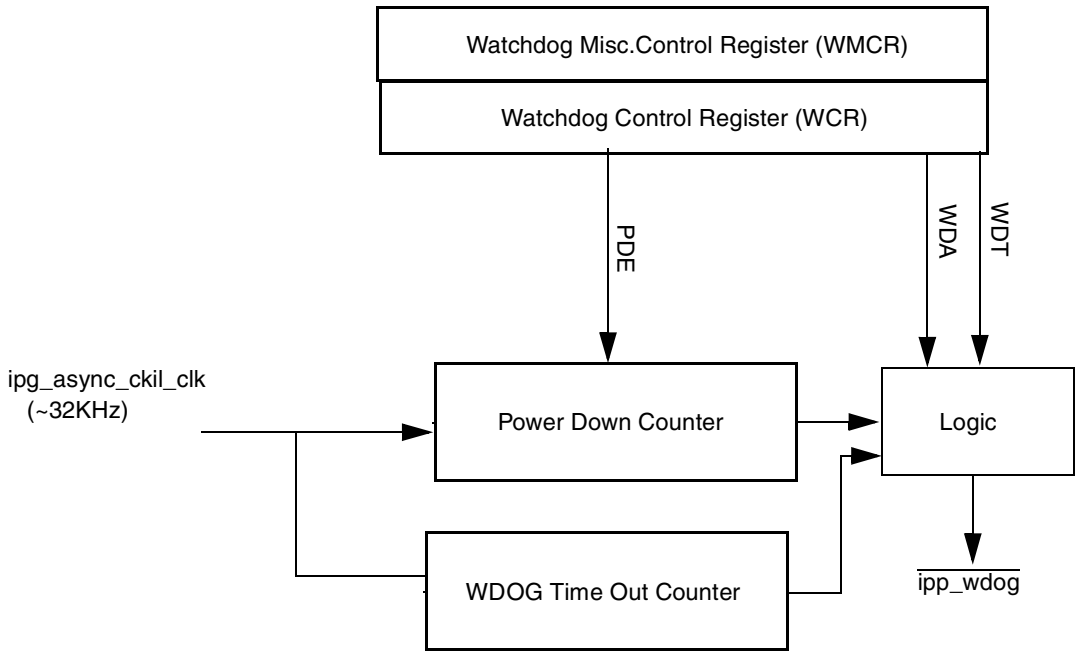


Figure 62-8. IPP_WDOG Operation

62.7 Initialization/Application Information

This section discusses the software restrictions and provides flow diagrams.

62.7.1 Software Restrictions

There are two software restrictions, as follows:

- The Power Down Counter, which is enabled out of any reset (Cold or Warm), time outs after 16 seconds. The ipp_wdog signal will be asserted low, and the WDI pin of the Power Management IC will be driven low. Therefore, once the system is out of reset, the software must ensure that it clears the PDE bit in WMCR within 16 seconds of reset (cold or warm) deassertion.
- The WDOG timer is not able to detect events that happen for periods of less than 1 ipg_clk_32k cycle. For example in repeated WAIT entry/exit, if the RUN mode time is less than 1 ipg_clk_32k cycle and if WDW bit is set, WDOG may not see any ipg_clk_32k edge during its wake time. Therefore, the WDOG timer may never time out although the system is in RUN mode for a finite duration. The time duration for events such as debug entry/exit should be more than 2 ipg_clk_32k cycles.

62.7.2 Flow Diagrams

Figure 62-9, Figure 62-10, and Figure 62-11 show the flow diagrams for the watchdog operation.

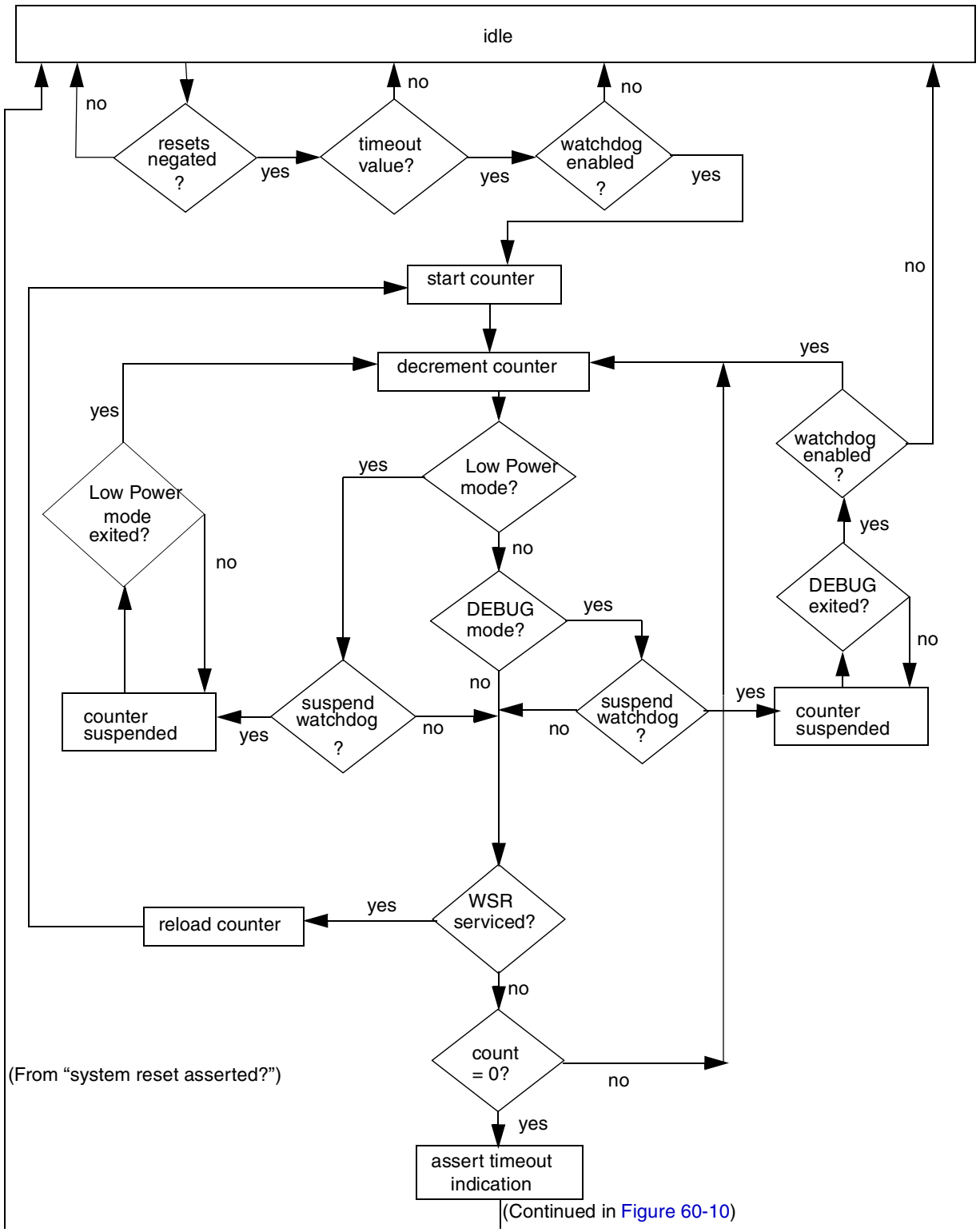
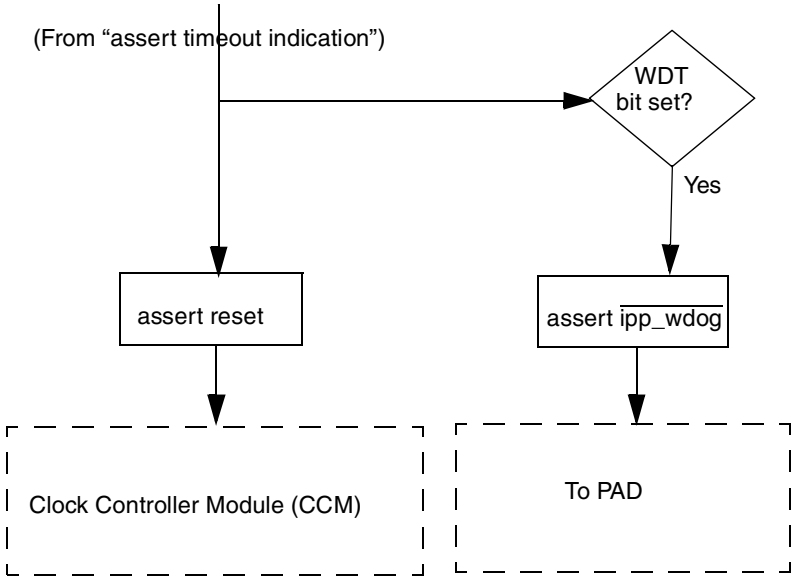


Figure 62-9. Time-out Counter Flow Diagram, 1of 2



NOTE: A system reset will force the state machine to “idle” at any time during countdown.

Figure 62-10. Time-out Counter Flow Diagram, 2 of 2

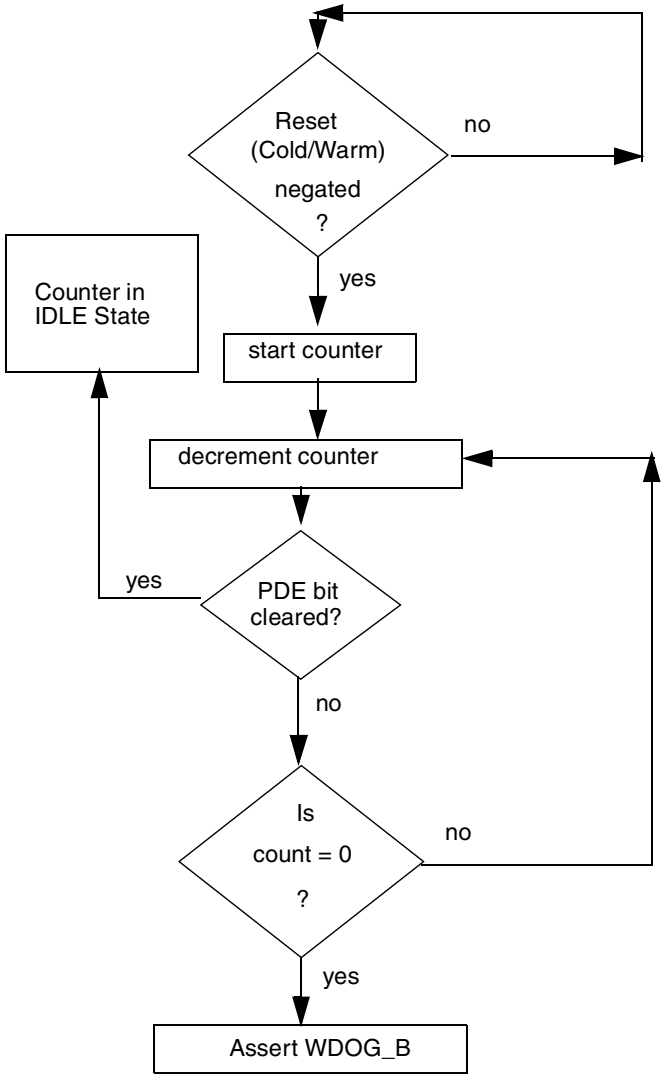


Figure 62-11. Power-Down Counter Flow Diagram

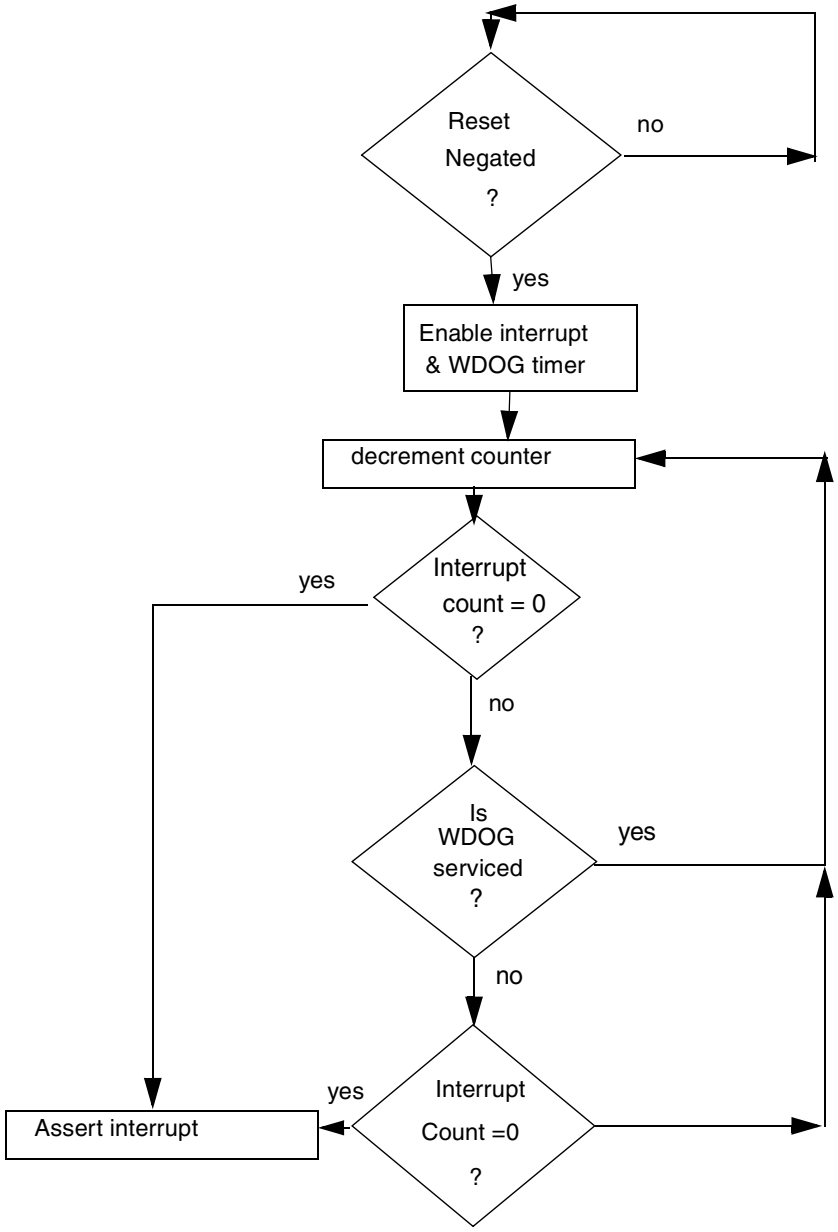


Figure 62-12. Interrupt Generation Flow Diagram

Chapter 63
Wireless External Interface Module (WEIM)

The Wireless External Interface Module (WEIM) handles the interface to devices external to the chip, including generation of chip selects, clock and control for external peripherals and memory. It provides asynchronous access to devices with SRAM-like interface and synchronous access to devices with NOR-Flash like or PSRAM like interface. Figure 63-1 illustrates a high level block diagram of the WEIM.

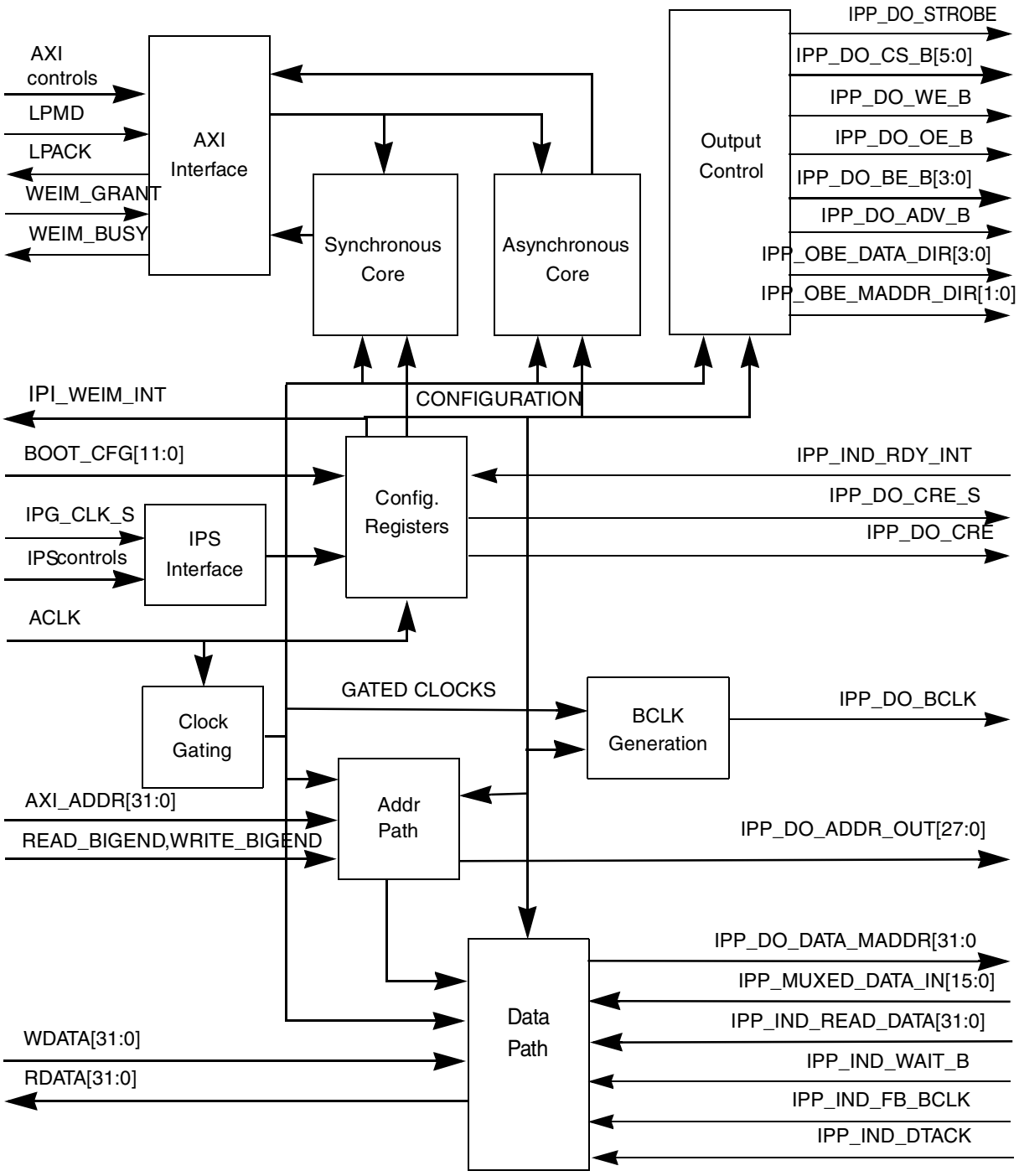


Figure 63-1. WEIM Block Diagram

63.1 Features

The following list describes the key features:

- Up to six chip selects for external devices:
  - Flexible address decoding. Each chip select memory space determine separately, according to VIA port configuration (see Section 63.4.1.1, Chip Select Memory Map"). Configurable Chip Select 0 base address (by VIA).
  - Individual select signal for each one of the memory space defined. Up to 6 memory spaces may be defined and program individually.
  - 28 bit external address bus, max memory size can be 256 Mbyte (2 Gigabit).
- Selectable Write Protection for each Chip Select
- Support for multiplexed address/data bus operation ×16 and ×32 port size
- Programmable Data Port Size for each Chip Select (×8, ×16, and ×32).
- Programmable Wait-State generator for each Chip Select, for write and read accesses separately.
- Asynchronous accesses with programmable setup and hold times for control signals
- Support for Asynchronous page mode accesses (×16 and ×32 port size).
- Independent synchronous Memory Burst Read Mode support for Nor-Flash and PSRAM memories (×16 and ×32 port size)
- Independent synchronous Memory Burst Write Mode support for PSRAM and Nor-Flash like memories (CellularRAM™ from Micron, Infineon, and Cypress, OneNand™ and utRAM™ from Samsung, and COSMORAM™ from Toshiba)
- Support of NAND Flash devices with a NOR Flash-like interface—MDOC™ (M-Systems), OneNand (Samsung)
- Independent programmable variable/fix Latency support for read and write synchronous (burst) mode.
- Support for Big Endian and Little Endian operation modes per access
- ARM AXI slave interface. One ID at a time support
- External Interrupt support, RDY_INT signal function as external interrupt
- Boot from external device support according to boot signals, using RDY_INT signal.
  - RDY signal support assertion after reset for MDOC (M-Systems) device
  - INT signal support assertion after reset for OneNand (Samsung) device

63.2 Modes of Operation

The WEIM has the following modes of operation:

- Asynchronous Mode
- Asynchronous Page Mode
- Multiplexed Address/Data mode
- Burst Clock Mode

- Low Power Modes
- Boot Mode

See details in Section 63.5.3, WEIM Operational Modes."

63.2.1 Asynchronous Mode

Asynchronous mode is a non-burst mode that is used for SRAM access. In this mode, a single data is read/written with each access (asserted address). All control timings are controlled by preset values in chip select configuration registers.

63.2.2 Asynchronous Page Read Mode

Setting the APR bit causes the WEIM to perform memory bursted accesses by emulating page mode operation. the external address asserts for each piece of data. The initial access timing is according to RWSC field, and the next address assertions timing is according to PAT field. When APR bit is set, RCSN OEN, RADVN, and RBEN fields are ignored for burst access to the external device.

The page size can be set via the BL field to 2,4, 8, 16, or 32 words (the word size is determined by the DSZ field).

63.2.3 Multiplexed Address/Data Mode

In this mode multiplexing addresses and data bits on the same pins is supported for synchronous/asynchronous accesses to ×8/×16/×32 data width memory devices. Table 63-1 shows the pins that drives data/address in 8/16/32 non-muxed mode and 16/32 muxed mode.

Table 63-1. Data and Address Bus

	Non Multiplexed Address/Data Mode						
	8 Bit (DSZ=100)	**8 Bit (DSZ=101)**	**8 Bit (DSZ=110)**	**8 Bit (DSZ=111)**	**16 Bit (DSZ=001)**	**16 Bit (DSZ=010)**	**32 bit (DSZ = 011)**
Address	ipp_do_addr_out	ipp_do_addr_out	ipp_do_addr_out	ipp_do_addr_out	ipp_do_addr_out	ipp_do_addr_out	ipp_do_addr_out
Data In	ipp_ind_ read_data [7:0]	ipp_ind_ read_data [15:8]	ipp_ind_ read_data [23:16]	ipp_ind_ read_data [31:24]	ipp_ind_ read_data [15:0]	ipp_ind_ read_data [31:16]	ipp_ind_ read_data
Data Out	ipp_do_ data_maddr [7:0]	ipp_do_ data_maddr [15:8]	ipp_do_ data_maddr [23:16]	ipp_do_ data_maddr [31:24]	ipp_do_ data_maddr [15:0]	ipp_do_ data_maddr [31:16]	ipp_do_ data_maddr
	Multiplexed Address/Data mode						
Address					ipp_do_addr_out		ipp_do_addr_out
Data In					ipp_muxed_data _in		ipp_ind_read_da ta
Data Out					ipp_do_addr_out [15:0]		ipp_do_data_ma ddr

63.2.4 Burst Clock Mode

The controller has the ability to support a burst synchronous operations in a various frequencies, depending on the frequency of the input clock supplied by the system (WEIM clock). The WEIM clock can be divided by one, Two, Three or four and it frequency can be change according to the requirements. Variable and fix latency are supported for this mode, according to the external device requirements.

- Synchronous read mode. This is a burst mode, it is used for reading from Flash/PSRAM memory devices. In this mode after address assertion a burst of sequential data can be read. Data exchange is carried out according to BCLK been generated by WEIM. An access is delayed according to external WAIT_B signal assertion (signal from the memory device).
- Synchronous write mode. A burst mode, it is used for accessing an external devices which support synchronous write type of access (PSRAM protocol). In this mode after address assertion a burst of sequential data can be written to the external device. Access may be delayed according to WAIT_B signal assertion (signal from the memory device) before first piece of data arrived to the external device.

NOTE

Maximum frequency of the WEIM main clock is 133 MHz. It may be reduced by the system for special cases of external devices, which demand a different frequency then integer division of the 133-MHz clock.

63.2.5 Low-Power Modes

The Input Clock is gated by ACT_CS bits, when all the ACT_CS are negated (all CS disable) the internal clock is turned off. Awready/wready and arready signal are de-asserted and the master cannot access the WEIM.

63.2.6 Boot Mode

It is possible to perform a boot operation from the external device located on CS0. The relevant bits are configured by boot mode signals according to the external device parameters (port size, protocol assertion, and so on). Table 63-2 shows the boot configuration settings.

Table 63-2. Boot Configuration Settings

BOOT_CFG Bits	Configured Bits	Place
11	DSZ[2]	CS0GCR1
10	AUS	CS0GCR1
9:8	CSREC[2:1]	CS0GCR1
7:5	RWSC[4:2], WWSC[4:2]	CS0RCR1, CS0WCR
4	ERRST	WCR
3	RAL,WAL	CS0RCR1,CS0WCR

Table 63-2. Boot Configuration Settings (continued)

BOOT_CFG Bits	Configured Bits	Place
11	DSZ[2]	CS0GCR1
2	MUM, OEA[1]	CS0GCR1, CS0RCR1
1:0	DSZ[1:0]	CS0GCR1

63.3 External Signal Description

This section provides an overview and detailed description of the signal properties.

63.3.1 Overview

Table 63-3 lists external signal properties.

Table 63-3. Signal Properties

Name	Port	Function	I/O	Reset	Pull Up
ACLK		AXI clock	I	Low	
ACLK_SLOW		AXI all time clock	I	Low	
IPG_CLK_S		IPG clock	I	Low	
RST_B		Reset	I		
WEIM_WARM_RESET		Warm Reset	I		
BOOT_CFG[11:0]		Boot Configuration	I	—	
WEIM_NFC_BASE[3:0]	VIA	WEIM CS0 base address	I	—	
ACT_CS[5:0]	VIA	Active chip selects configuration	I	—	
ADDRS0[1:0]	VIA	Address Space of CS0	I	—	
ADDRS1[1:0]	VIA	Address Space of CS1	I	—	
ADDRS2[1:0]	VIA	Address Space of CS2	I	—	
ADDRS3[1:0]	VIA	Address Space of CS3	I	—	
ADDRS4[1:0]	VIA	Address Space of CS4	I	—	
ADDRS5[1:0]	VIA	Address Space of CS5	I	—	
BIT_24_CRE_VIA	VIA	CRE is driven on bit 24	I		
BIT_25_CRE_VIA	VIA	CRE is driven on bit 25	I		
BIT_26_CRE_VIA	VIA	CRE is driven on bit 26	I		
BIT_27_CRE_VIA	VIA	CRE is driven on bit 27	I		
AWADDR[31:0]	AXI WAC	Axi Write Address	I	—	
AWSIZE[2:0]	AXI WAC	Axi write access size	I	—	
AWBURST[1:0]	AXI WAC	Axi write access burst type	I	—	

Table 63-3. Signal Properties (continued)

Name	Port	Function	I/O	Reset	Pull Up
AWLEN[3:0]	AXI WAC	Axi write access length	I	—	
AWPORT0	AXI WAC	Axi write access protection type	I	—	
AWVALID	AXI WAC	Axi write address valid	I	—	
AWID[3:0]	AXI WAC	Axi write address ID	I	—	
WRITE_BIGEND	AXI WAC	Axi write address endian mode	I	—	
AWREADY	AXI WAC	WEIM ready for write address	O	—	
WDATA[31:0]	AXI WDC	Axi Write Data	I	—	
WSTRB[3:0]	AXI WDC	Axi write byte strobe	I	—	
WID[1:0]	AXI WDC	Axi write data ID	I	—	
WALID	AXI WDC	Axi write data valid	I	—	
WLAST	AXI WDC	Axi last write data	I	—	
WREADY	AXI WDC	WEIM ready for write data	O	—	
BRESP	AXI WRC	Axi write response	O	—	
BVALID	AXI WRC	Axi write response valid	O	—	
BID	AXI WRC	Axi write response ID	O	—	
BREADY	AXI WRC	AXI ready for write response	I	—	
ARADDR[31:0]	AXI RAC	Axi read Address	I	—	
ARSIZE[2:0]	AXI RAC	Axi read access size	I	—	
ARBURST[1:0]	AXI RAC	Axi read access burst type	I	—	
ARLEN[3:0]	AXI RAC	Axi read access length	I	—	
ARPORT0	AXI RAC	Axi read access protection type	I	—	
ARVALID	AXI RAC	Axi read address valid	I	—	
ARID[3:0]	AXI RAC	Axi read address ID	I	—	
READ_BIGEND	AXI RAC	Axi read address endian mode	I	—	
ARREADY	AXI RAC	WEIM ready for read address	O	—	
RDATA[31:0]	AXI RDC	Axi read Data	O	—	
RID[1:0]	AXI RDC	Axi read data ID	O	—	
RVALID	AXI RDC	Axi read data valid	O	—	
RRESP	AXI RDC	Axi read error response	O	—	
RLAST	AXI RDC	Axi last read data	O	—	
RREADY	AXI RDC	AXI ready for read data	I	—	
IPS_ADDR	IPS	IPS address	I	—	
IPS_MODULE_EN	IPS	IPS module enable	I	—	

Table 63-3. Signal Properties (continued)

Name	Port	Function	I/O	Reset	Pull Up
IPS_RWB	IPS	IPS write enable	I	—	
IPS_WDATA[31:0]	IPS	IPS write data	I	—	
IPS_RDATA[31:0]	IPS	IPS read data	O	—	
IPS_XFR_ERR	IPS	IPS error response	O	—	
IPS_XFR_WAIT	IPS	IPS wait response	O	—	
IPI_WEIM_INT	IPS	Interrupt toward the master	O	—	
LPMD		Low power request	I		
LPACK		Low power acknowledge	O		
WEIM_GRANT		Handshake signal from other slaves or arbiter	I		
WEIM_BUSY		Handshake signal to other slaves or arbiter	O		
IPP_IND_BCLK	IPP	Burst Clock (BCLK)	O	0	
IPP_IND_RDY_INT	IPP	External interrupt/ready after reset for NandFlash Based memories with NorFlash interface	I	—	
IPP_IND_FB_BCLK	IPP	Feedback Burst Clock	I	0	
IPP_IND_READ_DATA[31:0]	IPP	Input Data Bus	I	—	
IPP_MUXED_DATA_IN[15:0]	IPP	Input Data Bus in muxed mode	I	—	
IPP_IND_WAIT_B	IPP	Memory Ready/Busy/Wait (WAIT)	I	—	pull up
IPP_DO_CS_B[5:0]	IPP	Chip Selects (CS)	O	High	
IPP_DO_OE_B	IPP	Memory Output Enable (OE)	O	1	
IPP_DO_WE_B	IPP	Memory Write Enable (WE)	O	1	
IPP_DO_BE_B[3:0]	IPP	Bytes Enable (BE)	O	High	
IPP_DO_ADV_B	IPP	Address Valid (ADV)	O	1	
IPP_DO_CRE	IPP	Memory Register Set	O	Low	
IPP_DO_CRE_SEL	IPP	CRE signal select	O	Low	
IPP_DO_STROBE	IPP	Logic analyzer signal	O	Low	
IPP_DO_ADDR_OUT[27:0]	IPP	Address Bus	O	Low	
IPP_DO_DATA_MADDR_OUT[31:0]	IPP	Output Data bus/Address and DATA Bus in Multiplexed mode	O	Low	
IPP_DO_DATA_DIR[3:0]	IPP	DATA IO Direction	O	Low	
IPP_DO_MADDR_DIR[1:0]	IPP	Muxed address IO Direction	O	Low	

63.3.2 Detailed Signal Descriptions

The following is a detailed description of the WEIM signals mentioned in Table 63-4.

Table 63-4. Detailed Signal Descriptions

Signal	I/O		Description
ACLK	I		AXI clock, maximal frequency 133 Mhz
ACLK_SLOW	I		AXI all time clock
IPG_CLK_S	I		IPG clock.
RST_B	I		Active low HW reset.
WEIM_WARM_RESET	I		Warm Reset. If this signal is asserted the rst_b will reset only the internal FF and state machine while SW registers will keep their current state.This signal is active high signal.
BOOT_CFG[11:0]	I		Boot Configuration. These input pins determine the state after reset of DSZ[2],AUS, CSREC[2:1],DSZ[1:0], MUM, OEA[1], RAL,RWSC[4:2] of CSRCR0 and ERRST bit of WCR., see Table 63-2 for detailed description.
WEIM_NFC_BASE[3:0]	I		WEIM CS0 base address. Those via connection are holding the 4 MSB of the WEIM base address. **Note:** In case those inputs are driven by fuses/external register and not by VIA the value of those signals can be changed only during reset. WEIM_NFC_BASE shouldn't be configured to F in order to avoid memory address wrap around.
ACT_CS[5:0]	I		Active chip selects. Each bit in the ACT_CS[5:0] bus represent one of the six chip selects of the WEIM. When ACT_CS[x]=1'b1, the corresponding chip select is active and has a valid address space according to it address space configuration determine from ADDRSx[1:0] bus input. The configuration is fixed and is set by VIA connection, please refer to the SOC chapter for the specific WEIM chip select configuration. **Note:** In case those inputs are driven by fuses/external register and not by VIA the value of those signals can be changed only during reset.
ADDRS0[1:0] ADDRS1[1:0] ADDRS2[1:0] ADDRS3[1:0] ADDRS4[1:0] ADDRS5[1:0]	I		Address Space. ADDRSx[1:0] is setting the space for each chip selects which is active (see ACT_CS[5:0] input description). The address space of the first active chip selects must be the biggest one, the following active chip selects address space may be equal or lower. Total address space size is 512 Mbyte. The configuration is fixed and is set by VIA connection, please refer to the SOC chapter for the specific WEIM chip select configuration. Please see Table for values supported and examples of configurations. **Note:** In case those inputs are driven by fuses/external register and not by VIA the value of those signals can be changed only during reset.
BIT_24_CRE_VIA			CRE is driven on bit 24
BIT_25_CRE_VIA			CRE is driven on bit 25
BIT_26_CRE_VIA			CRE is driven on bit 26
BIT_27_CRE_VIA			CRE is driven on bit 27
AWADDR[31:0]	I		Axi Write Address
AWSIZE[2:0]	I		Axi write access size
AWBURST[1:0]	I		Axi write access burst type.
AWLEN[3:0]	I		Axi write access length

Table 63-4. Detailed Signal Descriptions (continued)

Signal	I/O		Description
AWPORT0	I		Axi write access protection type. Only privileged/Normal mode are supported. 0 Normal Mode. 1 Privileged Mode.
AWVALID	I		Axi write address valid
AWID[3:0]	I		Axi write address ID
WRITE_BIGEND	I		Axi write address endian mode. 0 Access is in little endian. 1 Access is in big endian.
AWREADY	O		WEIM ready for write address
WDATA[31:0]	I		Axi Write Data
WSTRB[3:0]	I		Axi write byte strobe
WID[1:0]	I		Axi write data ID
WALID	I		Axi write data valid
WLAST	I		Axi last write data
WREADY	O		WEIM ready for write data
BRESP	O		Axi write response
BVALID	O		Axi write response valid
BID	O		Axi write response ID
BREADY	I		AXI ready for write response
ARADDR[31:0]	I		Axi read Address
ARSIZE[2:0]	I		Axi read access size
ARBURST[1:0]	I		Axi read access burst type
ARLEN[3:0]	I		Axi read access length
ARPORT0	I		Axi read access protection type
ARVALID	I		Axi read address valid
ARID[3:0]	I		Axi read address ID
READ_BIGEND	I		Axi read address endian mode
ARREADY	O		WEIM ready for read address
RDATA[31:0]	O		Axi read Data
RID[1:0]	O		Axi read data ID
RVALID	O		Axi read data valid
RRESP	O		Axi read error response
RLAST	O		Axi last read data
RREADY	I		AXI ready for read data

Table 63-4. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPS_ADDR[11:2]	I		IPS address
IPS_MODULE_EN	I		IPS module enable
IPS_RWB	I		IPS write enable. Active low signal. 0 - write 1- read
IPS_WDATA[31:0]	I		IPS write data
IPS_RDATA[31:0]	O		IPS read data
IPS_XFR_ERR	O		IPS error. Ips error will be generated whenever ips_module_en is asserted and ips_addr points to illegal address. (ips_addr[11:2] > 26'h)
IPS_XFR_WAIT	O		IPS wait. Ips wait will be asserted in case the access to register need more than 1 cycle. (see WEIM IP access register and Error address register).
IPI_WEIM_INT	O		Interrupt toward the master
LPMD	I		Low power request. Upon getting a low power request the WEIM completes the current AXI access, asserts the lpack signal and de-assert the wready/awready and arready signals. De-assertion of the lpmd will de-assert the lpack signal and assert the wready/awready and arready signals.
LPACK	O		Low power acknowledge. Lpack will be asserted if lpmd is high and there is no active/pending axi access.
WEIM_GRANT	I		Handshake signal from NFC. 0 - NFC is using the shared memory bus. Therefore, WEIM can't start axi access (if new access reaches WEIM it will be pending until WEIM_GRANT will be deasserted). 1- NFC isn't using the shared memory bus. Therefore, WEIM can access the bus immediately. **Note**: In 16 bit Muxed WEIM does not use the data bus therefore, there is no sharing of data bus with NFC and WEIM does not wait for grant from NFC.
WEIM_BUSY	O		Handshake signal to NFC. 0 WEIM does not have active or pending access or the current access is 16 Muxed mode. 1 WEIM has active or pending access and the current access is not 16 muxed mode.
IPP_DO_BCLK	O		Burst Clock (BCLK). This active-high output signal is used to clock external, burst-capable devices to synchronize the loading and incrementing of addresses and delivery of burst read and write data to/from the WEIM. Its behavior is affected by the BCM bit in the WEIM Configuration Register and the SWR, SRD, BCD and BCS bit fields in the Chip Select Configuration Registers.
IPP_IND_DTACK	I		Data Ack. for async. access. This input is used as a data ack. signal for single async. accesses.
IPP_IND_RDY_INT	I		External interrupt for NandFlash Based memories with NorFlash interface. ready after reset - Some External devices may use this signal for initial period after reset to indicate of a ready/wait state before boot sequence. This feature can be set by ERRST bit in the WCR.
IPP_IND_FB_BCLK	I		Burst Clock Feedback. This input is used to provide input data sampling clock in high data speed. It is a feedback from the IO PAD of the BCLK output pin that intend to align the clock used by the memory, and the one that is used to sample the read data.
IPP_MUXED_DATA_IN[15:0]	I		Input Data Bus. This bus is the 16 LSB input data from external devices in 32 bit muxed mode.

Table 63-4. Detailed Signal Descriptions (continued)

Signal	I/O	Description	
IPP_IND_READ_DATA[31:0]	I		Input Data Bus. This bus is the 32 bit input data from external devices (in 32 bit muxed mode only 16 LSB are used).
IPP_IND_WAIT_B	I		Ready/Busy/Wait signal. This active-low input signal is asserted by external burst capable devices which support fix or variable latency of data. It is serviced in synchronous mode only (SWR=1/SRD=1). WAIT will have a pull up resistor in IO. The signal indicates whether the External device is ready for data transaction or not. Busy cycles (or wait cycles) of the external device can occur at the start of a Burst access or at page boundary crossover. **Note:** For burst devices WAIT output should be configured to change one cycle before data is ready (before delay). Some External devices may not use this input signal for ready state indication (fix latency without WAIT signal monitoring), in this case WEIM should be configured accordingly (see RFL/WFL/PSZ bits description).
IPP_DO_CS_B[5:0]	O		Chip Selects. This signals are chip selects active-low output pins. Its behavior is affected by the RCSA/WCSA and RCSN/WCSN bit fields in the Chip Select Configuration Registers.
IPP_DO_OE_B	O		Output Enable. This active-low output signal indicates the bus access is a read and enables external devices to drive the data bus with read data. Its behavior is affected by the OEA and OEN bit fields in the Chip Select Configuration Registers.
IPP_DO_WE_B	O		Memory Write Enable. This active-low output signal indicates the bus access is a write and enables external devices to sample the data bus. Its behavior is affected by the WEA and WEN bit fields in the Chip Select Configuration Registers.
IPP_DO_BE_B[3:0]	O		Byte Enable. Those active-low output signals indicate active data bytes for the current access. They may be configured to assert for write cycles only. BE[0] corresponds to DATA_OUT[7:0], BE[1] corresponds to DATA_OUT[15:8], BE[2] corresponds to DATA_OUT[23:16], BE[3] corresponds to DATA_OUT[31:24]. n write accesses its behavior is affected by the BEA and BEN bit fields for Asynchronous mode only. In read access (synchronous or asynchronous) it always assert at start of access and negate at end of the access.
IPP_DO_ADV_B	O		Address Valid. This active-low output signal is asserted during burst mode accesses to cause the external burst capable device to load a new starting burst address. Assertion of ADV indicates that a valid address is present on the address bus. Its behavior is affected by the SWR, SRD, BCD, BCS, RADVA, WADVA, WAVDN, and RADVN bit fields in the Chip Select Configuration Registers. In asynchronous mode, ADV length is affected by the RADVA, WADVA, WADVN, and RADVN bit fields. Minimum length of ADV signal in all modes is one WEIM clock cycle.
IPP_DO_CRE	O		CRE is used as CRE/PS for CellularRam type of memory. It used for Mode Register Set command. It can be configured ad active low or active high signal. See CRE and CREP bits description in the CSGCRx registers.
IPP_DO_CRE_SEL	O		CRE select used for top level muxing of CRE signal with other signal
IPP_DO_STROBE	O		Strobe. This signal allows capture current access controls, address and data on logic analyzer. Sync. and async. accesses are supported. Note: Strobe signal for read data is active (RL + 1) cycles after data on external bus is valid.

Table 63-4. Detailed Signal Descriptions (continued)

Signal	I/O		Description
IPP_DO_ADDR_OUT[27:0]	O		Address Bus. These pins are used as address bits [27:0]. In 32 bit data multiplexed mode the 16 LSB of the data out and the 16 LSB of the address are muxed on the 16 LSB address bits.These pins represent [27:2] AXI address bits for word-width memory, [26:1] bits for half-word width memory and [25:0] bits for byte-width memory. **Note:** CRE bit is driven on one bit of ADDR_OUT[27:24] according to BIT_2x_CRE_VIA. In 32 bit muxed mode BE [3:2] are driven of ADDR_OUT[23:22]
IPP_DO_DATA_MADDR_OUT[31:0]	O		Output Data Bus. Those signals are output data bus used to transfer data to an external devices. Bidirectional data bus are made in IO PAD from DATA_IN[31:0] and DATA_OUT[31:0]. In 32 bit multiplexed mode the 16 MSB of data out and the 12 MSB of the address are muxed on the 16 LSB of the data. In 16 bit multiplexed mode this bus is not used. **Note**: In 32 bit muxed mode CRE bit is driven on DATA_MADDR_OUT[15] during address phase.
IPP_DO_DATA_DIR[3:0]	O		Input/Output directions for the DATA bus pads. 1 value indicates OUT direction, Zero value indicates Input direction (default). When working in multiplexed mode this bus influenced from ADH bit field in the chip select configuration registers.
IPP_DO_MADDR_DIR[1:0]	O		Input/Output directions for the address bus pads. 1 value indicates OUT direction, Zero value indicates Input direction (default). When working in multiplexed mode this bus influenced from ADH bit field in the chip select configuration registers. In non-muxed mode this directions will be always output.

63.4 Memory Map and Register Definition

The WEIM module includes 33 user-accessible 32-bit registers. The common register called the WEIM Configuration Register (WCR) which contains control bits that configure the WEIM for certain operation modes. Other 30 registers named Chip Select 0–5 General 1, General 2, Read and Write Configuration register correspondingly (CS0GCR1, CS0GCR2, CS0RCR1, CS0RCR2, CS0WCR,...,CS5GCR1, CS5GCR2, CS5RCR1, CS5RCR2, CS5WCR) and compose six groups of Chip Select Configuration Registers 0–5 (CS0CR–CS5CR) for each chip select. Configuration Registers layout is slightly different for the register group of CS0–CS0CR, because CS0CR reset state depends on boot signals input. These registers are accessible with IPS control signals only with word size (32-bit) reads and writes accesses. Additional registers are the WEIM IP access register (for ips sync. access and CS0 boot config. sync.) and Error address register (holds AXI error address).

63.4.1 Memory Map

Table 63-5 shows the WEIM memory map.

Table 63-5. WEIM Registers Memory Map

Address	Register	Access	Reset Value	Section/Page
General Registers				
0xBASE+0x2020 (CS0GCR1)	Chip Select 0 General Configuration Register 1	R/W[1]	0x00XY008W X-BOOT_CFG[10:8],0 Y-BOOT_CFG[11], BOOT_CFG[1:0],0 0, W-BOOT_CFG[2],0,0, 1	63.4.3.1/63-20
0xBASE+0x2024 (CS0GCR2)	Chip Select 0 General Configuration Register 2	R/W[2]	0x00000002	63.4.3.1/63-20
0xBASE+0x2008 (CS0RCR1)	Chip Select 0 Read Configuration Register 1	R/W	0xXY0WZ000 X-0,0,0,BOOT_CFG[7] Y-BOOT_CFG[6:5],0, 0 W-BOOT_CFG[3],0,1, 0 Z-0,0,BOOT_CFG[2], 0	63.4.3.3/63-25
0xBASE+0x200C (CS0RCR2)	Chip Select 0 Read Configuration Register 2	R/W	0x0000_0000	63.4.3.3/63-25
0xBASE+0x2010 (CS0WCR1)	Chip Select 0 Write Configuration Register 1	R/W	0x0009_2480	/63-30
0XBASE_2014 (CS0WCR2)	Chip Select 0 Write Configuration Register 2	RW	0x0000_0000	/63-33
0xBASE+0x2018 (CS1GCR1)	Chip Select 1 General Configuration Register 1	R/W	0x0001_0080	Figure 63-3
0xBASE+0x201C (CS1GCR2)	Chip Select 1 General Configuration Register 2	R/W	0x0000_0000	Figure 63-4
0xBASE+0x2020 (CS1RCR1)	Chip Select 1 Read Configuration Register 1	R/W	0x0000_0000	Figure 63-5
0xBASE+0x2024 (CS1RCR2)	Chip Select 1 Read Configuration Register 2	R/W	0x0000_0000	Figure 63-6
0xBASE+0x2028 (CS1WCR1)	Chip Select 1 Write Configuration Register 1	R/W	0x0000_0000	Figure 63-7
0XBASE_202C (CS1WCR2)	Chip Select 1Write Configuration Register 2	RW	0x0000_0000	Figure 63-8
0xBASE+0x2030 (CS2GCR1)	Chip Select 2 General Configuration Register 1	R/W	0x0001_0080	Figure 63-3

Table 63-5. WEIM Registers Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x2034 (CS2GCR2)	Chip Select 2 General Configuration Register 2	R/W	0x0000_0000	Figure 63-4
0xBASE+0x2038 (CS2RCR1)	Chip Select 2 Read Configuration Register 1	R/W	0x0000_0000	Figure 63-5
0xBASE+0x203c (CS2RCR2)	Chip Select 2 Read Configuration Register 2	R/W	0x0000_0000	Figure 63-6
0xBASE+0x2040 (CS2WCR1)	Chip Select 2 Write Configuration Register 1	R/W	0x0000_0000	Figure 63-7
0XBASE+0x2044 (CS2WCR2)	Chip Select 2 Write Configuration Register 2	RW	0x0000_0000	Figure 63-8
0xBASE+0x2048 (CS3GCR1)	Chip Select 3 General Configuration Register 1	R/W	0x0001_0080	Figure 63-3
0xBASE+0x204C (CS3GCR2)	Chip Select 3 General Configuration Register 2	R/W	0x0000_0000	Figure 63-4
0xBASE+0x2050 (CS3RCR1)	Chip Select 3 Read Configuration Register 1	R/W	0x0000_0000	Figure 63-5
0xBASE+0x2054 (CS3RCR2)	Chip Select 3 Read Configuration Register 2	R/W	0x0000_0000	Figure 63-6
0xBASE+0x2058 (CS3WCR1)	Chip Select 3 Write Configuration Register 1	R/W	0x0000_0000	Figure 63-7
0XBASE_205c (CS0WCR2)	Chip Select 3 Write Configuration Register 2	RW	0x0000_0000	Figure 63-8
0xBASE+0x2060 (CS4GCR1)	Chip Select 4 General Configuration Register 1	R/W	0x0001_0080	Figure 63-3
0xBASE+0x2064 (CS4GCR2)	Chip Select 4 General Configuration Register 2	R/W	0x0000_0000	Figure 63-4
0xBASE+0x2068 (CS4RCR1)	Chip Select 4 Read Configuration Register 1	R/W	0x0000_0000	Figure 63-5
0xBASE+0x206C (CS4RCR2)	Chip Select 4 Read Configuration Register 2	R/W	0x0000_0000	Figure 63-6
0xBASE+0x2070 (CS4WCR1)	Chip Select 4 Write Configuration Register 1	R/W	0x0000_0000	Figure 63-7
0XBASE_2074 (CS0WCR2)	Chip Select 4 Write Configuration Register 2	RW	0x0000_0000	Figure 63-8
0xBASE+0x2078 (CS5GCR1)	Chip Select 5 General Configuration Register 1	R/W	0x0001_0080	Figure 63-3
0xBASE+0x207C (CS5GCR2)	Chip Select 5 General Configuration Register 2	R/W	0x0000_0000	Figure 63-4
0xBASE+0x2080 (CS5RCR1)	Chip Select 5 Read Configuration Register 1	R/W	0x0000_0000	Figure 63-5

Table 63-5. WEIM Registers Memory Map (continued)

Address	Register	Access	Reset Value	Section/Page
0xBASE+0x2084 (CS5RCR2)	Chip Select 5 Read Configuration Register 2	R/W	0x0000_0000	Figure 63-6
0xBASE+0x2088 (CS5WCR1)	Chip Select 5 Write Configuration Register 1	R/W	0x0000_0000	Figure 63-7
0XBASE_208c (CS0WCR2)	Chip Select 5 Write Configuration Register 2	RW	0x0000_0000	Figure 63-8
0xBASE+0x2090 (WCR)	WEIM Configuration Register	R/W	0x0000_0020	Figure 63-9
0xBASE+0x2094 (WIAR)	WEIM IP Access Register	R/W	0x0000_001X X-BOOT_CFG[4],0,0,0	Figure 63-10
0xBASE+0x2098 (EAR)	Error Address Register	R/W	0x0000_0000	Figure 63-11

1 Note that R/W registers may contain some read-only or write-only bits.

2 Note that R/W registers may contain some read-only or write-only bits.

63.4.1.1 Chip Select Memory Map

Table 63-6 shows the chip selection memory map.

Table 63-6. Chip Selection Memory Map

Address	Space Size	Use	Access
According to WEIM_NFC_BASE /ACT_CS/ADDRS0 inputs	0 Mbyte–256 Mbyte	CS0 memory region	R/W
According to ACT_CS/ADDRS1 inputs	0 Mbyte–256 Mbyte	CS1 memory region	R/W
According to ACT_CS/ADDRS2 inputs	0 Mbyte–256 Mbyte	CS2 memory region	R/W
According to ACT_CS/ADDRS3 inputs	0 Mbyte–256 Mbyte	CS3 memory region	R/W
According to ACT_CS/ADDRS4 inputs	0 Mbyte–256 Mbyte	CS4 memory region	R/W
According to ACT_CS/ADDRS5 inputs	0 Mbyte–256 Mbyte	CS5 memory region	R/W

63.4.2 Register Summary

The conventions in Figure 63-2 and Table 63-7 serve as a key for the register summary and individual register diagrams.

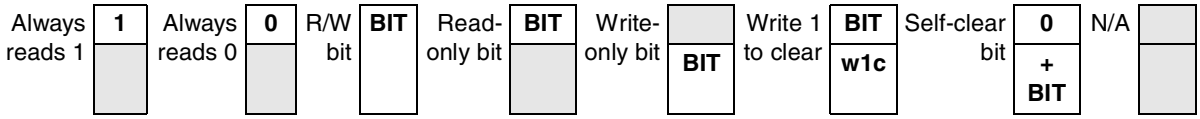


Figure 63-2. Key to Register Fields

Table 63-7 provides a key for register figures and tables and the register summary.

Table 63-7. Register Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writable.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
R	Read only. Writing this bit has no effect.
W	Write only.
R/W	Standard read/write bit. Only software can change the bit's value (other than a hardware reset).
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A status bit that can be read, and is cleared by writing a one.
Self-clearing bit	Writing a one has some effect on the module, but it always reads as zero. (Previously designated slfclr)
Reset Values	
0	Resets to zero.
1	Resets to one.
—	Undefined at reset.
u	Unaffected by reset.
[*signal_name*]	Reset value is determined by polarity of indicated signal.

Table 63-8 shows the register summary. Note that only the first entry reflects an actual register in this file. The other entries in the table diverge from the registers they summarize in order to display additional register summary examples.

Table 63-8. WEIM Register Summary

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x2020 (CS0GCR1)–0xBASE+0x2078 (CS5GCR1)	R	PSZ				WP	GBC			AUS	CSREC			SP	DSZ		
	W																
	R	BCS		BCD		WC	BL			CREP	CRE	RFL	WFL	MUM	SRD	SWR	CSEN
	W																
0xBASE+0x2024 (CS0GCR2)–0xBASE+0x207C (CS5GCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	DAP	DAE	DAPS				0	0	ADH	
	W																
0xBASE+0x2008 (CS0RCR1)–0xBASE+0x2080 (CS5RCR1)	R	0	0	RWSC						0	RADVA			RAL	RADVN		
	W																
	R	0	OEA			0	OEN			0	RCSA			0	RCSN		
	W																
0xBASE+0x200C (CS0RCR2)–0xBASE+0x2084 (CS5RCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	APR	PAT			0	0	RL		0	RBEA			RBE	RBEN		
	W																
0xBASE+0x2010 (CS0WCR1)–0xBASE+0x2088 (CS5WCR1)	R	WAL	WBED	WWSC						WADVA			WADVN			WBEA	
	W																
	R	WBEA	WBEN			WEA			WEN			WCSA			WCSN		
	W																
0xBASE+0x2018 (CS0WCR2)–0xBASE+0x208C (CS5WCR2)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	WBCDD
	W																
0xBASE+0x2090 (WCR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	WDOG_LIMIT		WDOG_EN	0	0	INTPOL	INTEN	0	GBCD		BCM
	W																

Table 63-8. WEIM Register Summary (continued)

Name		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0xBASE+0x2094 (WIAR)	R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																
	R	0	0	0	0	0	0	0	0	0	0	0	ACLK_EN	ERRST	INT	IPS_ACK	IPS_REQ
	W													R	R	R	
0xBASE+0x2098 (EAR)	R	Error Addr [31:16]															
	W	0															
	R	Error Addr [15:0]															
	W	0															

63.4.3 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number.

The 160 bits used to control Individual Chip Select are divided into five registers: Chip Select x General Configuration Register 1 (CSxGCR1), Chip Select x General Configuration Register 2 (CSxGCR2), Chip Select x Read Configuration register1 (CSxRCR1), Chip Select x Read Configuration register2 (CSxRCR2) and Chip Select x Write Configuration Register (CSxWCR). In addition there are three general registers WCR, WIAR, and EAR.

NOTE

All WEIM registers are sampled by IPG_CLK_S. Therefore IPG_CLK_S must be active when accessing through IP bus.

Read access from all registers (except WIAR and EAR) will generate 1 IPG_XFR_WAIT cycle.

Read access from WIAR and EAR will generates 6 IPG_XFR_WAIT cycles.

Write access to all registers (except EAR) will generates 3 IPG_XFR_WAIT cycles.

Write access to EAR will generates 6 IPG_XFR_WAIT cycles.

63.4.3.1 Chip Select x General Configuration Register 1

Figure 63-3 shows the chip select x general configuration register 1. Table 63-9 describes its fields.

Address 0xBASE+0x2020 (CS0GCR1)
0xBASE+0x2018 (CS1GCR1)
0xBASE+0x2030 (CS2GCR1)
0xBASE+0x2048 (CS3GCR1)
0xBASE+0x2060 (CS4GCR1)
0xBASE+0x2078 (CS5GCR1)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PSZ				WP	GBC			AUS	CSREC			SP	DSZ		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BCS		BCD		WC	BL			CREP	CRE	RFL	WFL	MUM	SRD	SWR	CSEN
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Figure 63-3. Chip Select x General Configuration Register 1

Table 63-9. Chip Select x General Configuration Register 1 Field Descriptions

Field	Description
31–28 PSZ	Page Size. This bit field indicates memory page size in words (word is defined by the DSZ field). PSZ is used when fix latency mode is applied, WFL=1 for sync. write accesses, RFL=1 for sync. Read accesses. When working in fix latency mode WAIT signal from the external device is not being monitored, PSZ is used to determine if page boundary is reached and renewal of access is preformed. This bit field is ignored when sync. Mode is disabled or fix latency mode is not being used for write or read access separately. It can be valid for both access type, read or write, or only for one type, according to configuration. PSZ is cleared by a hardware reset. 0000 8 words page size 0001 16 words page size 0010 32 words page size 0011 64 words page size 0100 128 words page size 0101 256 words page size 0110 512 words page size 0111 1024 (1k) words page size 1000 2048 (2k) words page size 1001–1111 Reserved
27 WP	Write Protect. This bit prevents writes to the address range defined by the corresponding chip select. WP is cleared by a hardware reset. 0 Writes are allowed in the memory range defined by chip. 1 Writes are prohibited. All attempts to write to an address mapped by this chip select result in a error response and no assertion of the chip select output.

Table 63-9. Chip Select x General Configuration Register 1 Field Descriptions (continued)

Field	Description
26–24 GBC	Gap Between Chip Selects. This bit field, according to the settings shown below, determines the minimum time between end of access to the current chip select and start of access to different chip select. GBC is cleared by a hardware reset. 000 minimum of 0 WEIM clock cycles before next access from different chip select (async. mode only) 001 minimum of 1 WEIM clock cycles before next access from different chip select 001 minimum of 2 WEIM clock cycles before next access from different chip select ... 111 minimum of 7 WEIM clock cycles before next access from different chip select
23 AUS	Address UnShifted. This bit indicates an unshifted mode for address assertion for the relevant chip select accesses. AUS bit is cleared by hardware reset. 0 Address shifted according to port size (DSZ config.) 1 Address unshifted **Note**: Reset value for CS0GCR1 for is BOOT_CFG[10]. For CS1GCR1-CS5GCR1 reset value is 0.
22–20 CSREC	CS Recovery. This bit field, according to the settings shown below, determines the minimum pulse width of CS, OE, and WE control signals before executing a new back to back access to the same chip select. CSREC is cleared by a hardware reset. 000 0 WEIM clock cycles minimum width of CS, OE and WE signals (read async. mode only) 001 1 WEIM clock cycles minimum width of CS, OE and WE signals 010 2 WEIM clock cycles minimum width of CS, OE and WE signals ... 111 7 WEIM clock cycles minimum width of CS, OE and WE signals **Note:** Reset value for CS0GCR1 for CSREC[2:1] is BOOT_CFG[9:8], for CSREC[0] is 0. For CS1GCR1-CS5GCR1 reset value is 000.
19 SP	Supervisor Protect. This bit prevents accesses to the address range defined by the corresponding chip select when the access is attempted in the User mode. SP is cleared by a hardware reset. 0 User mode accesses are allowed in the memory range defined by chip select. 1 User mode accesses are prohibited. All attempts to access an address mapped by this chip select in User mode results in an error response and no assertion of the chip select output.
18–16 DSZ	Data Port Size. This bit field defines the width of an external device's data port as shown below. 000 Reserved. 001 16 bit port resides on DATA[15:0] 010 16 bit port resides on DATA[31:16] 011 32 bit port resides on DATA[31:0] 100 8 bit port resides on DATA[7:0] 101 8 bit port resides on DATA[15:8] 110 8 bit port resides on DATA[23:16] 111 8 bit port resides on DATA[31:24] **Note:** Only async. access supported for 8 bit port. Reset value for CS0GCR1 for DSZ is BOOT_CFG[11],BOOT_CFG[1:0].For CS1GCR1-CS5GCR1 reset value is 001.

Table 63-9. Chip Select x General Configuration Register 1 Field Descriptions (continued)

Field	Description
15–14 BCS	Burst Clock Start. When SRD=1 or SWR=1,this bit field determines the number of WEIM clock cycles delay from start of access before the first rising edge of BCLK is generated. When BCD=0 value of BCS=0 results in a half clock delay after the start of access. For other values of BCD a one clock delay after the start of access is applied, not an immediate assertion. BCS is cleared by a hardware reset. 00 0 WEIM clock cycle additional delay 01 1 WEIM clock cycle additional delay 10 2 WEIM clock cycle additional delay 11 3 WEIM clock cycle additional delay
13–12 BCD	Burst Clock Divisor. This bit field contains the value used to program the burst clock divisor for BCLK generation. It is used to divide the internal WEIMbus frequency. BCD is cleared by a hardware reset. **Note:** For other then the mentioned below frequency such as 104 MHz, WEIM clock (input clock) should be adjust accordingly. 00 Divide WEIM clock by 1 01 Divide WEIM clock by 2 10 Divide WEIM clock by 3 11 Divide WEIM clock by 4
11 WC	Write Continuous. The WI bit indicates that write access to the memory are always continuous accesses regardless of the BL field value. WI is cleared by hardware reset. 0 Write access burst length occurs according to BL value. 1 Write access burst length is continuous.
10–8 BL	Burst Length. The BL bit field indicates memory burst length in words (word is defined by the DSZ field) and should be properly initialized for mixed wrap/increment accesses support. Continuous BL value corresponds to continuous burst length setting of the external memory device. For fix memory burst size, type is always wrap. In case not matching wrap boundaries in both the memory (BL field) and Master access on the current address, WEIM update address on the external device address bus and regenerates the access. BL is cleared by a hardware reset. When APR=1, Page Read Mode is applied, BL determine the number of words within the read page burst. BL is cleared by a hardware reset for CS0GCR1 – CS5GCR1. 000 4 words Memory wrap burst length (read page burst size when APR = 1) 001 8 words Memory wrap burst length (read page burst size when APR = 1) 010 16 words Memory wrap burst length (read page burst size when APR = 1) 011 32 words Memory wrap burst length (read page burst size when APR = 1) 100 Continuous burst length (2 words read page burst size when APR = 1) 101 Reserved 110 Reserved 111 Reserved
7 CREP	Configuration Register Enable Polarity. This bit indicates CRE memory pin assertion state, active-low or active-high, while executing a memory register set command to the external device (PSRAM memory type). CREP is set by a hardware reset. **Note:** Whenever PSRAM is connected the CREP value must be correct also for accesses where CRE is disabled. For Non-PSRAM memory CREP value should be 1. 0 CRE signal is active low. 1 CRE signal is active high.
6 CRE	Configuration Register Enable. This bit indicates CRE memory pin state while executing a memory register set command to PSRAM external device. CRE is cleared by a hardware reset. 0 CRE signal use is disable. 1 CRE signal use is enable.

Table 63-9. Chip Select x General Configuration Register 1 Field Descriptions (continued)

Field	Description
5 RFL	Read Fix Latency. This bit field determine if the controller is monitoring the WAIT signal from the External device connected to the chip select (handshake mode - fix or variable data latency) or if it start sampling data according to RWSC field, it only valid in synchronous mode. RFL is cleared by a hardware reset. When RFL=1 Burst access is terminated on page boundary and resume on the following page according to BL bit field configuration, because WAIT signal is not monitored from the external device. 0 the External device WAIT signal is being monitored, and it reflect the external data bus state 1 the state of the External devices is determined internally (Fix latency mode only)
4 WFL	Write Fix Latency. This bit field determine if the controller is monitoring the WAIT signal from the External device connected to the chip select (handshake mode - fix or variable data latency) or if it start data transfer according to WWSC field, it only valid in synchronous mode. WFL is cleared by a hardware reset. When WFL=1 Burst access is terminated on page boundary and resume on the following page according to BL bit field configuration, because WAIT signal is not monitored from the external device 0 the External device WAIT signal is being monitored, and it reflect the external data bus state 1 the state of the External devices is determined internally (Fix latency mode only)
3 MUM	Multiplexed Mode. This bit determines the address/data multiplexed mode for asynchronous and synchronous accesses for 8 bit, 16 bit or 32 bit devices (DSZ config. dependent). 0 Multiplexed Mode disable 1 Multiplexed Mode enable **Note:** Reset value for CS0GCR1 for MUM is BOOT_CFG[2].For CS1GCR1-CS5GCR1 reset value is 0.
2 SRD	Synchronous Read Data. This bit field determine the read accesses mode to the External device of the chip select. The External device should be configured to the same mode as this bit implicates. SRD is cleared by a hardware reset. 0 Read accesses are in Asynchronous mode 1 Read accesses are in Synchronous mode **Note**: Sync. accesses supported only for 16/32 bit port.
1 SWR	Synchronous Write Data. This bit field determine the write accesses mode to the External device of the chip select. The External device should be configured to the same mode as this bit implicates. SWR is cleared by a hardware reset. 0 Write accesses are in Asynchronous mode 1 Write accesses are in Synchronous mode **Note**: Sync. accesses supported only for 16/32 bit port.
0 CSEN	CS Enable. This bit controls the operation of the chip select pin. CSEN is set by a hardware reset for CSGCR0 to allow external boot operation. CSEN is cleared by a hardware reset to CSGCR1–CSGCR5. 0 Chip select function is disabled; attempts to access an address mapped by this chip select results in an error respond and no assertion of the chip select output 1 Chip select is enabled, and is asserted when presented with a valid access. **Note**: Reset value for CS0GCR1 for CSEN is 1. For CS1GCR1-CS1GCR5 reset value is 0.

63.4.3.2 Chip Select x General Configuration Register 2

Figure 63-4 shows the register; Table 63-10 shows its field descriptions.

Address 0xBASE+0x2024 (CS0GCR2)
0xBASE+0x201C (CS1GCR2)
0xBASE+0x2034 (CS2GCR2)
0xBASE+0x204C (CS3GCR2)
0xBASE+0x2064 (CS4GCR2)
0xBASE+0x207C (CS5GCR2)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DAP	DAE	DAPS				0	0	ADH	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-4. Chip Select x General Configuration Register 2

Table 63-10. Chip Select x General Configuration Register 2 Field Descriptions

Field	Description
31–10	Reserved
9 DAP	Data Acknowledge Polarity. This bit indicates DTACK memory pin assertion state, active-low or active-high, while executing an async access using DTACK signal from the external device. DAP is cleared by a hardware reset. 0 DTACK signal is active high 1 DTACK signal is active low
8 DAE	Data Acknowledge Enable. This bit indicates external device is using DTACK pin as strobe/terminator of an async. access. DTACK signal may be used only in asynchronous single read (APR=0) or write accesses. DTACK poling start point is set by DAPS bit field. polarity of DTACK is set by DAP bit field. DAE is cleared by a hardware reset. 0 DTACK signal use is disable 1 DTACK signal use is enable
7–4 DAPS	Data Acknowledge Poling Start. This bit field determine the starting point of DTACK input signal polling. DAPS is used only in asynchronous single read or write accesses. **Note:** Since DTACK is an async. signal the start point of DTACK signal polling is at least 3 cycles after the start of access. DAPS is cleared by a hardware reset. 0000 3 WEIM clk cycle between start of access and first DTACK check 0001 4 WEIM clk cycles between start of access and first DTACK check 0010 5 WEIM clk cycles between start of access and first DTACK check ... 0111 10 WEIM clk cycles between start of access and first DTACK check .... 1011 14 WEIM clk cycles between start of access and first DTACK check ... 1111 18 WEIM clk cycles between start of access and first DTACK check

Table 63-10. Chip Select x General Configuration Register 2 Field Descriptions (continued)

Field	Description
3–2	Reserved
1–0 ADH	Address hold time - This bit field determine the address hold time after ADV negation when mum = 1 (muxed mode). When mum = 0 this bit has no effect. For read accesses the field determines when the pads direction will be switched. 00 0 cycle after ADV negation 01 1 cycle after ADV negation 10 2 cycle after ADV negation 11 Reserved **Note:** Reset value for CS0GCR2 for ADH is 10. For CS1GCR2-CS5GCR2 reset value is 00.

63.4.3.3 Chip Select x Read Configuration Register 1

Figure 63-5 shows the register; Table 63-11 shows its field descriptions.

Address 0xBASE+0x2008 (CS0RCR1)
0xBASE+0x2020 (CS1RCR1)
0xBASE+0x2038 (CS2RCR1)
0xBASE+0x2050 (CS3RCR1)
0xBASE+0x2068 (CS4RCR1)
0xBASE+0x2080 (CS5RCR1)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	RWSC						0	RADVA			RAL	RADVN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	OEA			0	OEN			0	RCSA			0	RCSN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-5. Chip Select x Read Configuration Register 1

Table 63-11. Chip Select x Read Configuration Register 1 Field Descriptions

Field	Description
31–30	Reserved
29–24 RWSC	Read Wait State Control. This bit field programs the number of wait-states, according to the settings shown below, for synchronous or asynchronous read access to the external device connected to the chip select. When SRD=1 and RFL=0, RWSC indicates the number of burst clock (BCLK) cycles from the start of an access, before the controller can start sample data.Since WAIT signal can be asserted one cycle before the first data can be sampled, the controller starts evaluating the WAIT signal state one cycle before, this is referred as handshake mode or variable latency mode. When SRD=1 and RFL=1, RWSC indicates the number of burst clock (BCLK) cycles from the start of an access, until the external device is ready for data transfer, this is referred as fix latency mode. When SRD=0, RFL bit is ignored, RWSC indicates the asynchronous access length and the number of WEIM clock cycles from the start of access until the external device is ready for data transfer. RWSC is cleared by a hardware reset. 000000 Reserved 000001 RWSC value is 1 000010 RWSC value is 2 ... 000111 RWSC value is 7 001000 RWSC value is 8 001000 RWSC value is 9 ... 100000 RWSC value is 32 100001 RWSC value is 33 100010 RWSC value is 34 ... 111101 RWSC value is 61 111110 RWSC value is 62 111111 RWSC value is 63 **Note:** Reset value for CS0RCR1 for RWSC[4:2] is BOOT_CFG [7:5] for RWSC[5] and RWSC[1:0] is 0. For CG1RCR1-CS1RCR5 reset value is 000000.
23	Reserved
22–20 RADVA	ADV Assertion. This bit field determines when ADV signal is asserted for synchronous or asynchronous read modes according to the settings shown below. RADVA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of access and ADV assertion 001 1 WEIM clock cycles between beginning of access and ADV assertion 010 2 WEIM clock cycles between beginning of access and ADV assertion ... 111 7 WEIM clock cycles between beginning of access and ADV assertion
19 RAL	Read ADV Low. This bit field determine ADV signal negation time. When RAL=1, RADVN bit field is ignored and ADV signal will stay asserted until end of access. When RAL=0 negation of ADV signal is according to RADVN bit field configuration. RAL is configured at reset time with the BOOT_CFG[3] pin for CS0RCR1. RAL is cleared by a hardware reset for CS1RCR1 – CS5RCR1.

Table 63-11. Chip Select x Read Configuration Register 1 Field Descriptions (continued)

Field	Description
18–16 RADVN	ADV Negation. This bit field determines when ADV signal to memory is negated during read accesses. When SRD=1 (synchronous read mode), ADV negation occurs according to the following formula: (RADVN + RADVA + BCD + BCS + 1) WEIM clock cycles from start of access. When asynchronous read mode is applied (SRD=0) and RAL=0 ADV negation occurs according to the following formula: (RADVN + RADVA + 1) WEIM clock cycles from start of access. RADVN is cleared by a hardware reset. **Note**: Reset value for CS0RCR1 for RADVN is 2. For CS1RCR1 - CS5RCR1 reset value is 000. **Note**: This field should be configured so ADV negation will occur before the end of access. For ADV negation at the same time with the end of access user should RAL bit.
15	Reserved
14–12 OEA	OE Assertion. This bit field determines when OE signal are asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. OEA is cleared by a hardware reset. In muxed mode OE assertion occurs (OEA + RADVN + RADVA + ADH +1) WEIM clock cycles from start of access. 000 0 WEIM clock cycles between beginning of access and OE assertion 001 1 WEIM clock cycles between beginning of access and OE assertion 010 2 WEIM clock cycles between beginning of access and OE assertion ... 111 7 WEIM clock cycles between beginning of access and OE assertion **Note**: Reset value for CS0RCR1 is 000 if BOOT_CFG[2] is 0. If BOOT_CFG[2] is 1 the reset value for CS0RCR1 is 010. Reset value CS1RCR1 - CS5RCR1 is 000.
11	Reserved
10–8 OEN	OE Negation. This bit field determines when OE signal is negated during read cycles in asynchronous single mode only (SRD=0 and APR = 0), according to the settings shown below. This bit field is ignored when SRD=1. OEN is cleared by a hardware reset. 000 0 WEIM clock cycles between end of access and OE negation 001 1 WEIM clock cycles between end of access and OE negation 010 2 WEIM clock cycles between end of access and OE negation ... 111 7 WEIM clock cycles between end of access and OE negation
7	Reserved
6–4 RCSA	Read CS Assertion. This bit field determines when CS signal is asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. RCSA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of read access and CS assertion 001 1 WEIM clock cycles between beginning of read access and CS assertion 010 2 WEIM clock cycles between beginning of read access and CS assertion ... 111 7 WEIM clock cycles between beginning of read access and CS assertion

Table 63-11. Chip Select x Read Configuration Register 1 Field Descriptions (continued)

Field	Description
3	Reserved
2–0 RCSN	Read CS Negation. This bit field determines when CS signal is negated during read cycles in asynchronous single mode only (SRD=0 and APR = 0), according to the settings shown below. This bit field is ignored when SRD=1. RCSN is cleared by a hardware reset. 000 0 WEIM clock cycles between end of read access and CS negation 001 1 WEIM clock cycles between end of read access and CS negation 010 2 WEIM clock cycles between end of read access and CS negation ... 111 7 WEIM clock cycles between end of read access and CS negation

63.4.3.4 Chip Select x Read Configuration Register 2

Figure 63-6 shows the register; Table 63-12 shows its field descriptions.

Address 0xBASE+0x200C (CS0RCR2)
0xBASE+0x2024 (CS1RCR2)
0xBASE+0x203c (CS2RCR2)
0xBASE+0x2054 (CS3RCR2)
0xBASE+0x206C (CS4RCR2)
0xBASE+0x2084 (CS5RCR2)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	APR	PAT			0	0	RL			RBEA			RBE	RBEN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-6. Chip Select x Read Configuration Register 2

Table 63-12. Chip Select x Read Configuration Register 2 Field Descriptions

Field	Description
31–16	Reserved
15 APR	Asynchronous Page Read. This bit field determine the asynchronous read mode to the external device. When APR=0, the async. read access is done as single word (where word is defined by the DSZ field). when APR=1, the async. read access executed as page read. page size is according to BL field config., RCSN,RBEN,OEN and RADVN are being ignored. APR is cleared by a hardware reset for CS1GCR1 – CS5GCR1. **Note:** SRD=0 and MUM=0 must apply when APR=1

Table 63-12. Chip Select x Read Configuration Register 2 Field Descriptions (continued)

Field	Description
14–12 PAT	Page Access Time. This bit field is used in Asynchronous Page Read mode only (APR=1). the initial access is set by RWSC as in regular asynchronous mode. the consecutive address assertions width determine by PAT field according to the settings shown below. when APR=0 this field is ignored. PAT is cleared by a hardware reset for CS1GCR1 – CS5GCR1. 000 Address width is 2 WEIM clock cycles 001 Address width is 3 WEIM clock cycles 010 Address width is 4 WEIM clock cycles 011 Address width is 5 WEIM clock cycles 100 Address width is 6 WEIM clock cycles 101 Address width is 7 WEIM clock cycles 110 Address width is 8 WEIM clock cycles 111 Address width is 9 WEIM clock cycles
11–10	Reserved
9–8 RL	Read Latency. This bit field indicates cycle latency when executing a synchronous read operation. The fields holds the feedback clock loop delay in aclk cycle units. This field is cleared by a hardware reset. 00 Feedback clock loop delay is up to 1 cycle for BCD = 0 or 1.5 cycles for BCD != 0 01 Feedback clock loop delay is up to 2 cycle for BCD = 0 or 2.5 cycles for BCD != 0 10 Feedback clock loop delay is up to 3 cycle for BCD = 0 or 3.5 cycles for BCD != 0 11 Feedback clock loop delay is up to 4cycle for BCD = 0 or 4.5 cycles for BCD != 0
7	Reserved
6–4 RBEA	Read BE Assertion. This bit field determines when BE signal is asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. RBEA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of read access and BE assertion 001 1 WEIM clock cycles between beginning of read access and BE assertion 010 2 WEIM clock cycles between beginning of read access and BE assertion ... 111 7 WEIM clock cycles between beginning of read access and BE assertion
3 RBE	Read BE enable. This bit field determines if BE will be asserted during read access. 0 BE are disabled during read access. 1 BE are enable during read access according to value of RBEA and RBEN bit fields.
2–0 RBEN	Read BE Negation. This bit field determines when BE signal is negated during read cycles in asynchronous single mode only (SRD=0 and APR=0), according to the settings shown below. This bit field is ignored when SRD=1. RBEN is cleared by a hardware reset. 000 0 WEIM clock cycles between end of read access and BE negation 001 1 WEIM clock cycles between end of read access and BE negation 010 2 WEIM clock cycles between end of read access and BE negation ... 111 7 WEIM clock cycles between end of read access and BE negation

63.4.3.5 Chip Select x Write Configuration Register 1

Figure 63-7 shows the register; Table 63-13 shows its field descriptions.

Address 0xBASE+0x2010 (CS0WCR1)
0xBASE+0x2028 (CS1WCR1)
0xBASE+0x2040 (CS2WCR1)
0xBASE+0x2058 (CS3WCR1)
0xBASE+0x2070 (CS4WCR1)
0xBASE+0x2088 (CS5WCR1)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WAL	WBED	WWSC						WADVA			WADVN			WBEA	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WBEA	WBEN			WEA			WEN			WCSA			WCSN		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-7. Chip Select x Write Configuration Register 1

Table 63-13. Chip Select x Write Configuration Register 1 Field Descriptions

Field	Description
31 WAL	Write ADV Low. This bit field determine ADV signal negation time in write accesses. When WAL=1, WADVN bit field is ignored and ADV signal will stay asserted until end of access. When WAL=0 negation of ADV signal is according to WADVN bit field configuration. WAL is configured at reset time with the BOOT_CFG[3] pin for CSWCR0. WAL is cleared by a hardware reset for CSWCR1 – CSWCR5.
30 WBED	Write Byte Enable Disable. When asserted this bit prevent from IPP_DO_BE_B[x] to be asserted during write accesses. This bit is cleared by hardware reset.

Table 63-13. Chip Select x Write Configuration Register 1 Field Descriptions (continued)

Field	Description
29–24 WWSC	Write Wait State Control. This bit field programs the number of wait-states, according to the settings shown below, for synchronous or asynchronous write access to the external device connected to the chip select. When SWR=1 and WFL=0, WWSC indicates the number of burst clock (BCLK) cycles from the start of an access, before the memory can sample the first data.Since WAIT signal can be asserted one cycle before the first data can be sampled, the controller starts evaluating the WAIT signal state one cycle before, this is referred as handshake mode or variable latency mode. When SWR=1 and WFL=1, WWSC indicates the number of burst clock (BCLK) cycles from the start of an access, until the external device is ready for data transfer, this is referred as fix latency mode. When SWR=0, WFL bit is ignored, WWSC indicates the asynchronous access length and the number of WEIM clock cycles from the start of access until the external device is ready for data transfer. WWSC is cleared by a hardware reset. 000000 Reserved 000001 WWSC value is 1 000010 WWSC value is 2 000011 WWSC value is 3 ... 111111 WWSC value is 63 **Note**: Reset value for CS0WCR for WWSC[4:2] is BOOT_CFG [7:5], for WWSC[5] and WWSC[1:0] is 0. For CS1WCR - CS5WCR5 reset value is 000000.
23–21 WADVA	ADV Assertion. This bit field determines when ADV signal is asserted for synchronous or asynchronous write modes according to the settings shown below. WADVA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of access and ADV assertion 001 1 WEIM clock cycles between beginning of access and ADV assertion 010 2 WEIM clock cycles between beginning of access and ADV assertion ... 111 7 WEIM clock cycles between beginning of access and ADV assertion
20–18 WADVN	ADV Negation. This bit field determines when ADV signal to memory is negated during write accesses. When SWR=1 (synchronous write mode), ADV negation occurs according to the following formula: (WADVN + WADVA + BCD + BCS + 1) WEIM clock cycles. When asynchronous read mode is applied (SWR=0) ADV negation occurs according to the following formula: (WADVN + WADVA + 1) WEIM clock cycles. **Note**: Test value for CS0WCR for WADVN is 2. For CS1WCR–CS5WCR reset value is 000. This field should be configured so ADV negation will occur before the end of access. For ADV negation at the same time with the end of access user should WAL bit.
17–15 WBEA	BE Assertion. This bit field determines when BE signal is asserted during write cycles in async. mode only (SWR=0), according to the settings shown below. BEA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of access and BE assertion 001 1 WEIM clock cycles between beginning of access and BE assertion 010 2 WEIM clock cycles between beginning of access and BE assertion ... 111 7 WEIM clock cycles between beginning of access and BE assertion **Note**: Reset value for CS0WCR for WBEA is 2. For CS1WCR - CS5WCR reset value is 000.
14–12 WBEN	BE[3:0] Negation. This bit field determines when BE[3:0] bus signal is negated during write cycles in async. mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. BEN is cleared by a hardware reset. 000 0 WEIM clock cycles between end of access and WE negation 001 1 WEIM clock cycles between end of access and WE negation 010 2 WEIM clock cycles between end of access and WE negation ... 111 7 WEIM clock cycles between end of access and WE negation **Note**: Reset value for CS0WCR for WBEN is 2. For CS1WCR - CS5WCR reset value is 000.

Table 63-13. Chip Select x Write Configuration Register 1 Field Descriptions (continued)

Field	Description
11–9 WEA	WE Assertion. This bit field determines when WE signal is asserted during write cycles (synchronous or asynchronous mode), according to the settings shown below. This bit field is ignored when executing a read access to the external device. WEA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of access and WE assertion 001 1 WEIM clock cycles between beginning of access and WE assertion 010 2 WEIM clock cycles between beginning of access and WE assertion ... 111 7 WEIMclock cycles between beginning of access and WE assertion **Note**: Reset value for CS0WCR for WEA is 2. For CS1WCR - CS5WCR reset value is 000.
8–6 WEN	WE Negation. This bit field determines when WE signal is negated during write cycles in asynchronous mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. WEN is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of access and WE assertion 001 1 WEIM clock cycles between beginning of access and WE assertion 010 2 WEIM clock cycles between beginning of access and WE assertion ... 111 7 WEIM clock cycles between beginning of access and WE assertion **Note**: Reset value for CS0WCR for WEN is 2. For CS1WCR - CS5WCR reset value is 000.
5–3 WCSA	Write CS Assertion. This bit field determines when CS signal is asserted during write cycles (synchronous or asynchronous mode), according to the settings shown below.this bit field is ignored when executing a read access to the external device. WCSA is cleared by a hardware reset. 000 0 WEIM clock cycles between beginning of write access and CS assertion 001 1 WEIM clock cycles between beginning of write access and CS assertion 010 2 WEIM clock cycles between beginning of write access and CS assertion ... 111 7 WEIMclock cycles between beginning of write access and CS assertion
2–0 WCSN	Write CS Negation. This bit field determines when CS signal is negated during write cycles in asynchronous mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. WCSN is cleared by a hardware reset. 000 0 WEIM clock cycles between end of read access and CS negation 001 1 WEIM clock cycles between end of read access and CS negation 010 2 WEIM clock cycles between end of read access and CS negation ... 111 7 WEIM clock cycles between end of read access and CS negation

63.4.3.6 Chip Select x Write Configuration Register 2

Figure 63-8 shows the register; Table 63-14 shows its field descriptions.

Address 0xBASE+0x2018 (CS0WCR2)
0xBASE+0x202C (CS1WCR2)
0xBASE+0x2044 (CS2WCR2)
0xBASE+0x205C (CS3WCR2)
0xBASE+0x2074 (CS4WCR2)
0xBASE+0x208C (CS5WCR2)

Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	WBCDD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-8. Chip Select x Write Configuration Register 2

Table 63-14. Chip Select x Write Configuration Register 2 Field Descriptions

Field	Description
31–1	Reserved.
0 WBCDD	Write Burst Clock Divisor Decrement. If this bit is asserted and BCD value is 0 sync. write access will be preformed as if BCD value is 1.When this bit is negated or BCD value is not 0 this bit has no effect. This bit is cleared by hardware reset.

63.4.3.7 WEIM Configuration Register

Figure 63-9 shows the register; Table 63-15 shows its field descriptions.

Address 0xBASE+0x2090 (WCR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	WDOG_LIMIT		WDOG_EN	0	0	INTPOL	INTEN	0	GBCD		BCM
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Figure 63-9. WEIM Configuration Register

Table 63-15. WEIM Configuration Register Field Descriptions

Field	Description
31–11	Reserved
10–9 WDOG_LIMIT	Memory Watch Dog (WDog) cycle limit. This bit field determines the number of BCLK cycles (ACLK cycles in dtack mode) before the WDOG counter terminates the access and send an error response to the master. 00 128 BCLK cycles 01 256 BCLK cycles 10 512 BCLK cycles 11 1024 BCLK cycles
8 WDOG_EN	Memory WDog enable. This bit controls the operation of the WDOG counter that terminates the WEIM access. 0 Memory WDOG is Disabled 1 Memory WDog is Enabled
7–6	Reserved
5 INTPOL	Interrupt Polarity. This bit field determines the polarity of the external device interrupt. 0 External interrupt polarity is active low 1 External interrupt polarity is active high
4 INTEN	Interrupt Enable. When this bit is set the External signal RDY_INT as active interrupt. When interrupt occurs, INT bit at the WCR will be set and the WEIM_EXT_INT signal will be asserted correspondingly. This bit is cleared by a hardware reset. 0 External interrupt Disable 1 External interrupt Enable
3	Reserved

Table 63-15. WEIM Configuration Register Field Descriptions (continued)

Field	Description
GBCD 1–2	General Burst Clock Divisor. When BCM bit is set, this bit field contains the value used to program the burst clock divisor for Continuos BCLK generation. The other BCD bit fields for each chip select are ignored. It is used to divide the internal AXI bus frequency. When BCM=0 GBCD bit field has no influence. GBCD is cleared by a hardware reset. 00 Divide WEIM clock by 1 01 Divide WEIM clock by 2 10 Divide WEIM clock by 3 11 Divide WEIM clock by 4
0 BCM	Burst Clock Mode. This bit selects the burst clock mode of operation. It is used for system debug mode. BCM is cleared by a hardware reset. **Note:** The BCLK frequency in this mode is according to GBCD bit field. The BCLK phase is opposite to the WEIM clock in this mode if GBCD is 0. This bit should be used only in async. accesses. No sync access can be executed if this bit is set. 0 The burst clock runs only when accessing a chip select range with the SWR/SRD bits set. When the burst clock is not running it remains in a logic 0 state. When the burst clock is running it is configured by the BCD and BCS bit fields in the chip select Configuration Register. 1 The burst clock runs whenever ACLK is active (independent of chip select configuration)

63.4.3.8 WEIM IP Access Register

Figure 63-10 shows the register; Table 63-16 shows its field descriptions.

Address 0xBASE+0x2094 (WIAR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	ACLK_EN	ERRST	INT	IPS_ACK	IPS_REQ
W													R	R	R	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	Boot_cfg[4]	0	0	0

Figure 63-10. WEIM IP Access Register

Table 63-16. WEIM IP Access Register Field Descriptions

Field	Description
31–5	Reserved
4 ACLK_EN	ACLK enable. This bit gates the ACLK for the WEIM except from FFs that get ipg_aclk_s. After reset ACLK is enabled. 0 ACLK is disabled 1 ACLK is enabled

Table 63-16. WEIM IP Access Register Field Descriptions (continued)

Field	Description
3 ERRST	Enable READY After Reset. This bit indicates RDY_INT signal is used by an External Device on CS0. It used as a ready/busy status after reset of the device. ERRST is configured at reset time with the BOOT_CFG[4] pin. It is sticky bit, self cleared once RDY_INT signal assertion is detected. BOOT_CFG[4] pin will be ignored until hardware reset will be applied. When ERRST=1 the first fetch access from WEIM to the external device located on CS0 will be pending until RDY_INT signal indicate external device is ready, then WEIM will execute the access. 0 RDY_INT After Reset Disable 1 RDY_INT After Reset Enable **Note:** Reset value for ERRST is BOOT_CFG[4].
2 INT	Interrupt. This bit indicates interrupt assertion by an external device according to RDY_INT signal. When polling this bit, INT=0 indicates interrupt not occurred and INT=1 indicates assertion of the external device interrupt. This bit is cleared by a hardware reset.
1 IPS_ACK	IPS ACK. The WEIM is ready for ips access. There is no active AXI access and no new AXI access is accepted until this bit is cleared. This bit is cleared by the master after it completes the ips accesses. 0 Master can't access ips. 1 Master can access ips.
0 IPS_REQ	IPS request. The Master requests to access one of the IPS registers.Dutring such access the WEIM should not preform any AXI/memory accesses. The WEIM finishes the AXI accesses that already starts and asserts the IPS_ACK bit. 0 No Master requests ips access 1 Master requests ips access

63.4.3.9 Error Address Register

Figure 63-11 shows the register; Table 63-17 shows its field descriptions.

Address 0xBASE+0x2098 (EAR) Access: User read-write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Error_Addr [31:16]															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Error_Addr [15:0]															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 63-11. Error Address Register

Table 63-17. Error Address Register Field Descriptions

Field	Description
31–0	Error Address. This bit field holds the AXI address of the last access that caused error. This register is read only register.

63.5 Functional Description

This section provides the functional description for the WEIM module.

63.5.1 Clocks

The WEIM has the following four input clocks:

- ACLK—WEIM clock (main clock, AXI clock), with a Max frequency of 133 MHz. This clock can be gated externally when there is no active AXI access.
- ACLK_SLOW—WEIM all time running ACLK. Used for FF that must be active even when WEIM is in low power down mode to provide clock for lpack/lpmd registers, IP registers, IP to AXI sync. registers.
- IPG_CLK_S—IPG clock for IP accesses.

NOTE

IP registers are activated by ACLK_SLOW clock.

BCLK is a clock which created from WEIM clock for External device usage. Integer division by 1, 2, 3, and 4 of the clock can be use with BCD bit field configuration, according to external devices demands.WEIM clock frequency may be reduce for lower frequency support which can’t be achieved via BCD bit field.

- FB_CLK is BCLK after pads delays.

63.5.2 Bus Sizing Configuration

The WEIM supports byte, half word, and word operands allowing access to ×8, ×16, ×32 ports. It can be address/data multiplexed in ×16, ×32 ports. The port size is programmable via the DSZ bit field in the corresponding Chip Select Configuration Register. A 8-bit port can reside in each one of the bytes of the data bus.A 16-bit port can reside on the lower 16 bits of the data bus, DATA_IN/OUT[15:0] or on the higher 16 bits of the data bus, DATA_IN/OUT[31:16] (see Table 63-1).

In the case of a multi-cycle transfer, the lower two address bits (ADDR[1:0]) are incremented appropriately. The WEIM address bus is configured according to DSZ bit field and AUS bits. There is either one bit (for ×16 port size) or two bits (for ×32 port size) right shift of the address bits (only when AUS=0) and no bit shift when AUS = 1 or DSZ[2] = 1.

The WEIM has a data multiplexer which takes the four bytes of the AXI data bus and routes them to their required positions to properly interface to memory.

NOTE

A word access to or from a ×16 port requires two external bus cycles to complete the transfer.

A word access to or from a ×8 port requires four external bus cycles to complete the transfer.

63.5.2.1 8-Bit Port Support

WEIM has limited support for mot68000 and Intel 386 protocols.

63.5.2.1.1 MOTOROLA 68000

WEIM has limited support for the mot6800 protocol; it only supports basic read or write asynchronous operation.

WEIM does not support the following for the mot6800 protocol:

- Read modify write operations
- Synchronous access operations
- All special accesses (CPU space, bus arbitration, bus control, bus error, and reset operations)
- FC outputs

63.5.2.1.2 Intel 386

WEIM has limited support for Intel 386 protocol; it only supports basic read or write asynchronous non-pipelined operations.

WEIM does not support the following for the Intel 386 protocol:

- Other bus cycle operations (interrupt, halt, and refresh)
- Bus Lock operation
- M/IO, DC, LBA, NA, REFRESH, and BS8 signals

63.5.3 WEIM Operational Modes

WEIM has the following main operational modes, which are selected by control bit fields settings. For details see the corresponding bit field descriptions of SWR/SRD/MUM. All modes are supported with 8-bit, 16-bit, or 32-bit port configuration, according to the DSZ bit field.

Table 63-18. WEIM Operation Modes Field Settings

Control Bit Fields			Brief Mode Description
MUM	SRD	SWR	
0	0	0	Asynchronous write/Asynchronous read for APR=0/ Asynchronous page read for APR=1, none multiplexed
		1	Synchronous write/ Asynchronous read or APR=0/ Asynchronous page read for APR=1,none multiplexed
	1	0	Asynchronous write/Synchronous read none multiplexed
		1	Synchronous write/read none multiplexed

Table 63-18. WEIM Operation Modes Field Settings (continued)

Control Bit Fields			Brief Mode Description
MUM	SRD	SWR	
1	0	0	Asynchronous write/read multiplexed
		1	Synchronous write/ Asynchronous read multiplexed
	1	0	Asynchronous write/Synchronous read multiplexed
		1	Synchronous write/read multiplexed

63.5.4 Burst Mode (Synchronous) Memory Operation

This mode is enabled for read or write access individually. Bit SWR set the burst mode for write operations at the corresponding chip select and bit SRD set it for read operation.

When this mode is set, the controller attempts to translate the Master burst accesses to memory burst accesses, being limited by the memory burst length, predefined by BL value, or memory and Master WRAP/INCR boundary crossing non-matching. Only the first address accessed is displayed by the controller on the external address bus, in a memory burst sequence.

WEIM may translate from some Master sequential accesses to one or few memory bursts, but not from two Master individual accesses to one memory burst.

For the first access in a memory burst sequence, the WEIM assert $\overline{\text{ADV}}$, causing the external burst device to latch the starting burst address and then toggle the burst clock (BCLK) a predefined number of cycles in order to latch the first unit of data. Subsequent accessed data units can then be burst in fewer clock cycles, realizing an overall increase in bus bandwidth.

NOTE

The BCLK signal toggles only when burst access is executed toward the external device (BCM=1’b0 for normal mode use), it runs with a 50% duty cycle until the end of access is reached. When access is terminated BCLK stop toggling.

Memory burst accesses are terminated by the WEIM whenever it detects the following:

- The specific burst length has executed completely—end of access
- Write access—missing data in write buffer (master is delaying the data transfer toward the WEIM
- Next sequential access crosses boundary with unequal condition (wrap/increment, burst length) on the Master and memory.
- Current memory burst length reached.

63.5.5 Burst Clock Divisor (BCD)

In some cases it may be necessary to slow the external bus in relation to the internal bus to allow accesses to burst devices that have a maximum operating frequency which is less than the operating frequency of

the internal bus. The internal bus frequency can be divided by one, two, three, or four for presentation on the external bus in burst mode operation.

To achieve frequency of BCLK other then an integer division, the cycle time of the WEIM clock entering the controller need to be changed accordingly.

By programming the BCD bit field to various values, two signals on the external bus are affected; $\overline{\text{ADV}}$ and BCLK. The $\overline{\text{ADV}}$ signal is asserted according to RADVA or WADVA bit fields programming and it negated according to the formula mentioned in RADVN and WADVN bit fields description. The BCLK signal runs with a 50% duty cycle until the end of access is reached.

If BCM = 1, the BCLK runs at frequency according to GBCD bit field settings on every async. Memory access (regardless of the SWR and SRD bits configuration). Caution should be exercised when using BCM bit, the GBCD bit field should be updated once and not change when BCLK is toggling. The BCM bit is used mainly for system debug mode. It has no functional use of the WEIM in normal mode.

63.5.6 Burst Clock Start (BCS)

In an effort to allow greater flexibility in achieving the minimum number of wait states on burst accesses, the user can determine when they want the BCLK to start toggling after the start of access. This allows the BCLK to be skewed from point of data capture on the WEIM clock by any number of WEIM clock cycles. Care must be exercised when setting BCS bit field in conjunction with the BCD, and RWSC/WWSC bit fields. See the external timing diagrams from Section 63.8.7, Burst Read Memory Accesses Timing Diagram - BCD=0," and Section 63.8.8, Burst Read Memory Accesses Timing Diagram - BCD=1," for some examples of when to use the BCS, BCD, and RWSC/WWSC bit fields together.

63.5.7 Multiplexed Address/Data Mode

A control bit MUM allows support memory with multiplexed address/data bus both in asynchronous and in synchronous modes. Caution should be exercised for using OEA/WEA and ADH bit fields. They should be configured according to the external device requirements, as it determines the time point of end of address phase and start of data phase.

63.5.8 Mixed Master/Memory Burst Modes Support

To provide mixed sequential/wrap accesses with different length WEIM interprets burst signal and generate additional $\overline{\text{ADV}}$ signals whenever there appear unequal address or burst boundary crossing condition. BL bit field is used to notify WEIM about current memory burst and wrap condition for properly external address generation. In case of non matching boundaries in both the memory and Master access, WEIM starts a new memory burst access by updating address from Master on address bus and generating $\overline{\text{ADV}}$ signal.

63.5.9 AXI (Master) Bus Cycles Support

The WEIM use an ARM AXI slave interface. It has a 32-bit bus and supports one access (one ID) at a time, no out of order or parallel accesses are supported.

The following AXI protocol signals are not supported: AWLOCK, AWCACHE, ARLOCK, and ARCACHE.

Bus sampling and signal use is as follows:

- ARID bus is sampled when new read access is valid on the read address channel and is reflected on the RID bus output toward the master.
- AWID bus is sampled when new write access is valid on the write address channel and is reflected on the WID/BID bus output toward the master.
- ARPROT and AWPROT signal are partially used. ARPROT[0] and AWPROT[0] bits are used for normal/privileged access detection. ARPROT[2:1] and AWPROT[2:1] are not used.
- When sampling a valid access on both of the address channels, the read access will be performed first while write access is pending. After last data transfer completed, the pending write will be executed.

A new access may be executed one cycle after sampling a valid access on the read or write address channels. For write access data should be present at write buffer for fast execution. This will apply assuming there is no current access (back to back) which can cause a recovery or end of access penalty cycles, according to configuration.

NOTE

Only 32-bit word size accesses are supported for burst mode accesses.

Only 8-bit (1 byte), 16-bit (2 byte) and 32-bit (4 byte) word size supported for single access.

Maximum number of burst length is 16.

According to AXI protocol burst access should not cross 4 KB blocks. In case WEIM gets an access that crosses the 4 Kbyte, memory address calculation is invalid.

AXI transfers are also supported as shown in Table 63-19. These AXI cycles will be translated into the necessary cycles on the memory side. For example for optimal operation in case ARM cache is configured to 8 beat burst with wrap, a synchronous flash and cellular RAM memory should be configured in 16 word wrap burst mode when using a 16-bit data port, and in 8 word wrap burst mode when using a 32-bit data port. WEIM uses BL bit field for support different memory configurations. The controller splits the transaction when needed in some cases as shown in Table 63-19.

Table 63-19. AXI Burst Cycles Supported

Burst Length—Number of Data Transfers	Burst Size—Bytes in Transfer	Burst Type	Description
1	1	INCR	Single transfer
1	2	INCR	Single transfer
1	4	INCR	Single transfer
2	4	WRAP	2-beat wrapping burst
4	4	WRAP	4-beat wrapping burst

Table 63-19. AXI Burst Cycles Supported (continued)

Burst Length—Number of Data Transfers	Burst Size—Bytes in Transfer	Burst Type	Description
8	4	WRAP	8-beat wrapping burst
16	4	WRAP	16-beat wrapping burst
2	4	INCR	2-beat incrementing burst
3	4	INCR	3-beat incrementing burst
4	4	INCR	4-beat incrementing burst
5	4	INCR	5-beat incrementing burst
6	4	INCR	6-beat incrementing burst
7	4	INCR	7-beat incrementing burst
8	4	INCR	8-beat incrementing burst
9	4	INCR	9-beat incrementing burst
10	4	INCR	10-beat incrementing burst
11	4	INCR	11-beat incrementing burst
12	4	INCR	12-beat incrementing burst
13	4	INCR	13-beat incrementing burst
14	4	INCR	14-beat incrementing burst
15	4	INCR	15-beat incrementing burst
16	4	INCR	16-beat incrementing burst

Table 63-20. AXI To Memory Burst Splits Number

AXI Burst Type	Memory Burst Type Config.	# of Accesses to X8 Memory Port size	# of Accesses to X16 Memory Port size	# of Accesses to X32 Memory Port size
INC16 Aligned Addr.	WRAP4	16	8	4
	Cont.	1	1	1
INC16 Unaligned Addr.	WRAP4	17	9	5
	Cont.	1	1	1
WRAP16 Aligned Addr.	WRAP16	4	2	1
	Cont.	1	1	1
WRAP16 Unaligned Addr.	WRAP16	5	3	1
	Cont.	2	2	2
INC8 Aligned Addr.	WRAP8	4	2	1
	WRAP16	2	1	1

Table 63-20. AXI To Memory Burst Splits Number (continued)

AXI Burst Type	Memory Burst Type Config.	# of Accesses to X8 Memory Port size	# of Accesses to X16 Memory Port size	# of Accesses to X32 Memory Port size
INC8 Unaligned Addr.	WRAP8	4 or 5	2 or 3	2
	WRAP16	2 or 3	2	1 or 2
WRAP8 Aligned Addr.	WRAP16	2	1	1
	Cont.	1	1	1
WRAP8 Unaligned Addr.	WRAP16	2 or 3	1	2
	Cont.	2	2	2

63.5.10 WAIT_B Signal, RWSC, and WWSC Bit Fields Usage

Most of the external devices supporting burst mode for write or read accesses provide a signal that indicates data is valid on the memory bus. For this mode, called handshake mode, the RFL and WFL bits are cleared and the RWSC/WWSC bit fields indicate when the controller should start sampling this signal from the external device, in other words, how many BCLK cycles should be masked.

For devices that are not using this signal or have a fixed latency ability, the RFL and WFL bits may be set for internal calculation regarding BCLK cycles penalty until data is valid (memory initial access time). For this mode, RWSC/WWSC indicates when the data is ready for sampling by the controller (read access) or the external device (write access). There is a separation between the read and write accesses wait-state control. For read access, the RWSC bit field is valid and the WWSC bit field is ignored whereas for write access, WWSC is valid and RWSC is ignored.

63.5.11 IPS Register Interface

Access to the registers of the WEIM, read or write, is made with IPS protocol signals. The system should avoid changing the registers while master/memory transaction is valid, as this can cause an unknown behavior of the controller.

The register access size is 32 bit as the register size definition. Other access sizes (byte or half word) are not supported.

63.5.12 MRS Set for PSRAM

Memory registers of PSRAM devices can be configured according to external signal, which indicates whether the access is to a memory array or memory register domain. When the CRE bit is set, the following transactions to the external device will assert the CRE signal. The polarity of this signal is determined by the CREP bit for active low or active high assertion of the signal.

63.5.13 WEIM Access Termination

WEIM is monitoring the corresponding CSx control signal every time variable latency access or dtack access is performed toward the external device.

In variable latency accesses the watch dog timer (counter) counts BCLK cycle if it reaches the wdog_limit (according the WDOG_LIMIT bit field in the WCR) before the device signals in can drive/sample new data the controller will terminate the access and generate an error response transfer toward the master.

In dtack access the watch dog counts ACLK cycles instead of BCLK and it reaches the wdog_limit before the device asserts the dtack signal the controller will terminate the access and generate an error response transfer toward the master.

This WDOG can by disabled by WDOG_EN bit in the WCR.

63.5.14 Error Conditions

The following conditions cause an error (AXI error or IPS error) response signal:

- AXI errors
  - Access to a disabled chip select - access to a mapped chip select address space where the CSEN bit in the corresponding chip select Configuration Register is clear
  - Access to a non mapped address - access to an address that is not mapped to any CS.
  - User access to a supervisor-protected chip select address space (the SP bit in the corresponding chip select Configuration Register is set)
  - User access in fixed mode access
  - User preform write access to write protected chip select
  - First write data ID and write address ID do not match. (No data is written to the memory).
  - First Write Data ID and write address ID match but one or more of the other Write data IDs does not match the First Write data ID (data is written to memory according)
  - Access duration to external device from CSx signal assertion is 128/256/512/1024 cycles (access is terminated by the controller) - This error can be disabled be software.
- IPS errors
  - User read or write access to a reserved/non-valid address in the WEIM Configuration Register

63.5.15 DTACK Mode

In DTACK mode the WEIM uses DTACK signal as an indication when to end the access.

DTACK is an asynchronous edge/level sensitive signal. DTACK polarity is configurable by DAP bit in CsxGCR2 (default value is 0).

In this case WEIM begins the access and after few cycles (according DAPS field) and waits until DTACK (after synchronization) becomes asserted then sample the data in read access and completes the current data access. (see Figure 63-25, Figure 63-26, and Figure 63-27).

If more than one data is needed CS will be negated between access (CSREC field is not zero) the AXI burst access will be split in to single accesses (see Figure 63-28).

63.5.16 RDY_INT Signal as Interrupt

The WEIM has an external interrupt support. When INTEN bit in the WCR is set, signal RDY_INT is used as interrupt, its status is being reflected by INT bit and output signal. It cleared by writing one to the INT bit. When INTEN is cleared, the interrupt is disable. This interrupt is a level interrupt and its polarity can be configured by INTPOL bit in the WCR.

63.5.17 RDY_INT Signal as Ready After Reset Indication

This feature is used for boot propose from external devices based on NandFlash array memory with NorFlash interface.

When ERRST bit is set. RDY_INT signal is monitored to determine ready after reset of the external device located on CS0.

The monitoring is taking place when CS0 is accessed for the first time. The access will be pending until assertion of the signal is detected. When detection occurs, the ERRST bit is self-cleared and pending access is executed to the external device on CS0.

63.5.18 WEIM_GRANT/WEIM_BUSY Handshake Description

Prior to executing command to one of the external device (chip select),WEIM assert WEIM_BUSY signal (1’b1) and check the WEIM_GRANT signal status. If WEIM_GRANT signal is high, it indicates external data bus is not used by other slaves (Nand Flash Controller) and WEIM may start to execute the access. If WEIM_GRANT is low, WEIM waits until it is set (1’b1) before executing the access.

WEIM keeps WEIM_BUSY signal set until it completes the access toward the external device.

Once WEIM_GRANT signal is set, it cannot be reset until WEIM_BUSY signal is cleared by the WEIM.

NOTE

The 16-bit Muxed WEIM does not use the data bus, therefore, there is no sharing of the data bus with NFC. WEIM does not wait for a WEIM_GRANT signal from NFC and does not assert the WEIM_BUSY signal.

63.5.19 LPMD/LPACK Handshake Description

These signals used for frequency and/or voltage change and entering low power mode during normal operation of the WEIM. Before any change is taking place, the controller and all the relevant external devices should be in idle state which means no access or data transfer is in process.

LPMD input signal is asserted, once WEIM detect the assertion of LPMD, all ready signals of the AXI channels are negated and WEIM is not sampling new accesses. It finish all the on going accesses and

already pending ones. When WEIM is in idle state LPACK output signal is asserted. WEIM will stay in idle state and LPACK signal will stay assert until LPMD signal is negated.

63.5.20 Endianness

Big- and little-endianness are supported by the controller according to the Table 63-21.

Table 63-21. WEIM Out/in Data in Case AXI Out/in Data is 0xB3B2B1B0

Endian mode	AXI access	AXI address [1:0]	Port Size and Used Bits								
			Word Port				Half Word Port			Byte Port	
			[31:24]	[23:16]	[15:8]	[7:0]	External address [0]	[31:24] ([15:8])	[23:16] ([7:0])	External address [1:0]	[31:24] ([23:16]) ([15:8]) ([7:0])
Big	Word	0	**0xB3**	**0xB2**	**0xB1**	**0xB0**	0	**0xB3**	**0xB2**	0	**0xB3**
										1	**0xB2**
							1	**0xB1**	**0xB0**	2	**0xB1**
										3	**0xB0**
	Half Word	0			**0xB1**	**0xB0**	0	**0xB3**	**0xB2**	0	**0xB3**
										1	**0xB2**
		2	**0xB3**	**0xB2**			1	**0xB1**	**0xB0**	2	**0xB1**
										3	**0xB0**
	Byte	0				**0xB0**	0		**0xB3**	0	**0xB3**
		1			**0xB1**			**0xB2**		1	**0xB2**
		2		**0xB2**			1		**0xB1**	2	**0xB1**
		3	**0xB3**					**0xB0**		3	**0xB0**

Table 63-21. WEIM Out/in Data in Case AXI Out/in Data is 0xB3B2B1B0 (continued)

Endian mode	AXI access	AXI address [1:0]	Port Size and Used Bits								
			Word Port				Half Word Port			Byte Port	
			[31:24]	[23:16]	[15:8]	[7:0]	External address [0]	[31:24] ([15:8])	[23:16] ([7:0])	External address [1:0]	[31:24] ([23:16]) ([15:8]) ([7:0])
Little	Word	0	**0xB3**	**0xB2**	**0xB1**	**0xB0**	0	**0xB1**	**0xB0**	0	**0xB0**
										1	**0xB1**
							1	**0xB3**	**0xB2**	2	**0xB2**
										3	**0xB3**
	Half Word	0			**0xB1**	**0xB0**	0	**0xB1**	**0xB0**	0	**0xB0**
										1	**0xB1**
		2	**0xB3**	**0xB2**			1	**0xB3**	**0xB2**	2	**0xB2**
										3	**0xB3**
	Byte	0				**0xB0**	0		**0xB0**	0	**0xB0**
		1			**0xB1**			**0xB1**		1	**0xB1**
		2		**0xB2**			1		**0xB2**	2	**0xB2**
		3	**0xB3**					**0xB3**		3	**0xB3**

63.5.21 Strobe Signal Use

The strobe signal is toggling according to address/data valid condition on the external bus for read and write accesses and for both synchronous and asynchronous modes.

At any time point when address/data is valid on the external bus, the strobe signal will generate a positive edge which can be used to sample the external data and control signal.

NOTE

Strobe signal for read data is active (RL + 1) cycles after data on external bus is valid.

63.6 Initialization Information

This section provides intialization information.

63.6.1 Active Chip Selects and Address Space Configuration

Each chip select activation is set using VIA port configuration. Refer to the SOC chapter for the specific WEIM chip select configuration. Chip select is determine when the corresponding bit of the ACT_CS[5:0] input is set to one, else it is in none active state. Each one of the six chip selects' address space can be

determined by ADDRS0[1:0]...ADDRS5[1:0] WEIM inputs. The minimum address space size per chip select is 32 Mbyte, maximum size is 256 Mbyte. Overall address space for all chip select is 512 Mbyte. Base address is {WEIM_NFC_BASE,28'h0}, topmost base address is CS0 base address + 0x1FFF_FFFF.

Address space order is determine from the largest space on CS0 to the smallest one on CS5. Not all the chip select must be used (determine by the active bit per CS inputs). Equal size for several chip select may apply. It is the user responsibility to ensure the total space size configured is within the 512 Mbyte dedicated to the WEIM.

NOTE

If the total address space is smaller then 512 Mbyte, it is referred as reserved space and can not be used.

Table 63-22. Examples of Address Space Configuration

Chip Select #	Example1	Example2	Example3	Example4
CS0	256 Mbyte	128 Mbyte	128 Mbyte	256 Mbyte
CS1	128 Mbyte	128 Mbyte	64 Mbyte	64 Mbyte
CS2	64 Mbyte	128 Mbyte	64 Mbyte	64 Mbyte
CS3	64 Mbyte	64 Mbyte	32 Mbyte	32 Mbyte
CS4	—	32 Mbyte	32 Mbyte	32 Mbyte
CS5	—	32 Mbyte	32 Mbyte	32 Mbyte

The address space is set with the matching ADDRSx[1:0] input bus according to Table 63-23.

Table 63-23. Address Space Configuration

ADDRSx[1:0]	Size
2'b00	32 Mbyte
2'b01	64 Mbyte
2'b10	128 Mbyte
2'b11	256 Mbyte

63.6.2 Booting from WEIM

WEIM is ready to work with CS0 after hardware reset, but it has been configured for very slowly access (for boot purpose) with additional setup and hold time. Other CS are disabled by hardware reset. Therefore, any CS has to be properly initialized before using it by writing values to the corresponding chip select configuration registers.

APR,PAT,BL,AUS, CSREC[2:1], DSZ, MUM, RWSC[4:2], RAL, WWSC[4:2], WAL and ERRST bit fields can be determined for reset values according to BOOT_CFG[11:0] pin settings.

63.7 Application Note

Application note uses next functions to illustrate WEIM and memory accesses, as follows:

- WR16(address, data) is a 16 bit write access
- WR32(address, data) is a 32 bit write access
- RD16(address, data) is a 16 bit read access
- RD32(address, data) is a 32 bit read access,
- WR_I(address, data, delta, counter) is a write data sequence, there data(i+1) = data(i)+ delta, COMMAND_SEQUENCE, CHECK_STATUS

All addresses are byte address. ‘CS0 is a Chip Select 0 base address. ‘WEIM_ is a prefix of WEIM’s registers. ‘h is a prefix of hexadecimal constant. // is a comment beginning. csba[cs] is a dimension of CS base addresses. “addr” means an address offset in current CS address space. Examples use CS0 address space, but it may be any except boot mode functionality. Configuration examples were checked with some below defined memory models and may require some adjustment for other family members.

63.7.1 Access to AMD Flash

The configurations in this section are for am29pdl128g_70.

63.7.1.1 Boot

To boot from AMD flash next boot, use the following configuration:

- `BOOT_CFG= ‘h263` for 32 bit memory
- `BOOT_CFG= ‘h261` for 16 bit memory

63.7.1.2 AMD Asynchronous Mode Configuration

The next WEIM register configuration used for AMD flash asynchronous access is as follows:

```
        WR32(‘WEIM_CS0RCR1,’h0a018000);
        WR32(‘WEIM_CS0WCR1,’h0704a240);
        WR32(‘WEIM_CS0GCR1,’h00430081); // for 32 bit memory
    or
        WR32(‘WEIM_CS0GCR1,’h00410081); // for 16 bit memory
```

63.7.1.3 AMD Synchronous Mode Configuration

63.7.1.4 Utility

Use the following configuration for utility:

```
    // Single data word programming to addr, 32-bit memory
        port_size = 32;
        WR32(‘CS0+(’h0555<<2),’h00aa); // command sequence
        WR32(‘CS0+(’h02aa<<2),’h0055);
        WR32(‘CS0+(’h0555<<2),’ha0); // command code
        WR32(‘CS0+addr, data); // addr & data
// Polling end of programming
```

```
    edata = data; // expected data
    data = ~edata;
    errst = 0;
while(!(data == edata) &&!errst)
       begin: BR_EN
       RD16(‘CS0+addr, data);          // Read status
       if(data[7]!== edata[7])
          begin
          if(data[5] == 1)
              begin
              RD16(‘CS0+addr, data3);
              RD16(‘CS0+addr, data);
              if(data[6] !== data3[6])
                  begin
                  $display("CHECK_STATUS: Error timout on single data program");
                  errst = 1;
                  disable BR_EN;
                  end
              end
         end
       else
          begin
          RD16(‘CS0+addr, data3);
          if(port_size == 32)
              RD32(‘CS0+addr, data);
          else
              begin
              RD16(‘CS0+addr, data);
              edata[31:16] = 16’h0;
              end
          if(data !== edata)
              begin
              $display("CHECK_STATUS: Error in data write on single data program");
                errst =2;
              disable BR_EN;
              end
          end
       end
// Single data word programming to addr, 16-bit memory
    port_size = 16;
    WR32(‘CS0+(’h0555<<1),’h00aa); // command sequence
    WR32(‘CS0+(’h02aa<<1),’h0055);
    WR32(‘CS0+(’h0555<<1),’ha0); // commund code
    WR32(‘CS0+addr, data); // addr & data
// Polling end of programming - the same as for 32 bit
```

63.7.2 Access to Intel Sibley Flash

The configurations in this section are for the Sibley family muxed and non-muxed devices.

63.7.2.1 Boot

To boot from Intel Sibley flash next boot use the following configuration:

- `BOOT_CFG= ‘h281` for 16 bit non-muxed memory,
- `BOOT_CFG= ‘h285` for 16 bit muxed memory.

63.7.2.2 Intel Sibley Flash Asynchronous Mode Configuration

Use the following configuration for Intel Sibley Flash asynchronous mode configuration.

```
WR32('WEIM_CS0GCR1,'h00210081);
WR32('WEIM_CS0RCR1,'h0e020000);
WR32('WEIM_CS0RCR2,'h00000000);
WR32('WEIM_CS0WCR1,'h0704a040);
```

63.7.2.3 Intel Sibley Flash Synchronous Mode Configuration

Use the following configuration for 133 MHz synchronous access to flash:

```
      // Set memory to synchronous read mode
           WR16('CS0+('h5903<<1),'h0060);
           WR16('CS0+('h5903<<1),'h0003);
           WR16('CS0+('h0000<<1),'h00ff);
      // Set WEIM configuration to synchronous timing
           WR32('WEIM_CS0GCR1,'h50214225);                    // 133 MHz
           WR32('WEIM_CS0RCR1,'h0c000000);                    // 12 cycles on memory
Next configuration used for 66 MHz synchronous access to muxed flash:
      // Set memory to synchronous read mode
           WR16('CS0+('h3103<<1),'h0060);
           WR16('CS0+('h3103<<1),'h0003);
           WR16('CS0+('h0000<<1),'h00ff);
      //-------------------------------------------------------------------------
      // Set WEIM configuration to synchronous timing
           WR32('WEIM_CS0GCR1,'h5021122d);                    // 66 MHz
           WR32('WEIM_CS0RCR1,'h07000000);                    // 7cycles on memory
```

63.7.2.4 Utility

The Intel Sibley device uses the following utility algorithms:

```
      // Single data word programming to addr
           WR16('CS0+addr,'h0060);                // Unlock
           WR16('CS0+addr,'h00d0);
           WR16('CS0+addr, 'h0041);
           WR16('CS0+addr, data);
           WR16('CS0+caddr, 'h0070);                // Read Status command
           while('CS0+data[7] == 0)                          // Wait / Polling
               RD16('CS0+addr, data);             // Read status
           RD16('CS0+addr, data);                 // Read status
           WR16('CS0+'h0000,'h00ff);
      // Write buffer programming
           WR16('CS0+addr,'h0060);                // Unlock
           WR16('CS0+addr,'h00d0);
           data    = 0;
           WR16('CS0+addr, 'h0070);               // Read Status command
           while(data[7] == 0)                           // Wait
               RD16('CS0+addr, data);             // Read status
WR16('CS0+'h0000,'h00ff);
           WR16('CS0+addr,'h00e9);          // Write Buffer command
           WR16('CS0+addr,255);             // Word counter (<256)
           for(i=0; i<'h200; i = i + 'h40)
              WR_I('CS0+addr+i, data+((i>2)*'h0010_0001), 'h0010_0001, 16); // Data
           WR16('CS0+addr,'h00d0);          // Write Confirm command
           data  = 0;
```

```
while(data[7] == 0)                    // Wait
   RD16(‘CS0+addr, data);          // Read status
RD16(‘CS0+addr, data);             // Read status
WR16(‘CS0+’h0000,’h00ff);
```

63.7.3 Access to MDOC Device

The configurations in this section are for the MDOC H3 device.

63.7.3.1 Boot

To boot from MDOC device, the ERRST bit should be configured to 1, so WEIM will hold the first reaccess to CS0 until the MDOC asserts the RDY signal.

- `BOOT_CFG= ‘h271` may be used for boot from non-muxed MDOC device
- `BOOT_CFG= ‘h275` may be used for boot from muxed MDOC device

63.7.3.2 MDOC Asynchronous Mode Configuration

Use the following for MDOC asynchronous mode configuration.

```
// Non-muxed mode
WR32(‘WEIM_CS0GCR1,’h00410081);
WR32(‘WEIM_CS0RCR1,’h0e121010);
WR32(‘WEIM_CS0RCR2,’h00000000);
WR32(‘WEIM_CS0WCR1,’h12092492);
// Muxed mode
WR32(‘WEIM_CS0GCR1,’h00410081);
WR32(‘WEIM_CS0RCR1,’h0e121010);
WR32(‘WEIM_CS0RCR2,’h00000000);
WR32(‘WEIM_CS0WCR1,’h12092492);
```

63.7.3.3 MDOC Synchronous Mode Configuration

TBD

63.7.3.4 Utility

The MDOC H3 device uses the following utility algorithms:

```
// Read Manufacturer ID and Device ID
    RE16(‘CS0+’h9400,’h4833);
    RE16(‘CS0+’h9422,’hb7cc);
```

63.7.4 Access to Micron PSRAM

The configurations in this section are for mt45w4mw16bfb_706.

63.7.4.1 Micron PSRAM Asynchronous Mode Configuration

Use the following for Micron PSRAM asynchronous mode configuration.

```
// 16 bit memory
           WR32('WEIM_CS0GCR1,'h403104b1);
           WR32('WEIM_CS0RCR1,'h0b010000);
           WR32('WEIM_CS0RCR2,'h00000008);
           WR32('WEIM_CS0WCR1,'h0b040040);
       // 32 bit memory
           WR32('WEIM_CS0GCR1,'h403304b1);
           WR32('WEIM_CS0RCR1,'h0f010000);
           WR32('WEIM_CS0RCR2,'h00000008);
           WR32('WEIM_CS0WCR1,'h0f040040);
```

63.7.4.2 Micron PSRAM Synchronous Mode Configuration

Use the following for Micron PSRAM synchronous mode configuration.

```
// 16 bit memory
           WR32('WEIM_CS0GCR1,'h403104b1);
           WR32('WEIM_CS0WCR1,'h0b040000);
           WR16('CS0+('h85947<<1),'h0040); // memory configuration
           WR32('WEIM_CS0GCR1,'h4021_5487);// fixed latency memory wrap 4
           WR32('WEIM_CS0RCR1,'h04000000);
           WR32('WEIM_CS0RCR2,'h00000008);
           WR32('WEIM_CS0WCR1,'h04000000);
       // 32 bit memory
           WR32('WEIM_CS0GCR1,'h6003_04f1);
           WR32('WEIM_CS0WCR1,'h0b04_0000);
           WR32('CS0+('h85947<<2),'h0040); // memory configuration
           WR32('WEIM_CS0GCR1,'h4003_1487); // var latency memory inc. page size 128
           WR32('WEIM_CS0RCR1,'h04000000);
           WR32('WEIM_CS0RCR2,'h00000008);
           WR32('WEIM_CS0WCR1,'h04000000);
```

63.7.5 Access to Samsung OneNAND

The configurations in this section are for kfg1g16q2a and kfm1g16q2a83.

63.7.5.1 Boot

There are two ways to boot from Samsung OneNand. In the first way, the ERRST bit set to 0 and user has to poll the interrupt status in the OneNand interrupt register (or set interrupt handler there). In the second way, the ERRST bit set to 1 and the user should enable the device interrupt output before the first read from CS0 access issued.

- `BOOT_CFG= 'h261` may be used for boot from non-muxed Samsung OneNand.
- `BOOT_CFG= 'h265` may be used for boot from muxed Samsung OneNand.

Load sectors 2,3 to DataRAM, page 0 as shown below:

```
WR16(‘CS0+(’hF241<<1),’h0);        // Clear interrupt status
WR16(‘CS0+(’hF100<<1),’h0);        // block[8:0] address
WR16(‘CS0+(’hF107<<1),’h2);        // sector[1:0] and page[7:2] addresses
WR16(‘CS0+(’hF200<<1),’h802);      // buffer[11:8] address and counter[1:0]
WR16(‘CS0+(’hF101<<1),’h0);        // DDP choose
WR16(‘CS0+(’hF220<<1),’h0);        // Set command
```

63.7.5.2 Samsung OneNand Asynchronous Mode Configuration

Use the following for Samsung OneNand asynchronous mode configuration.

```
// Non-muxed memory
    WR32(‘WEIM_CS0GCR1,’h00410081);
    WR32(‘WEIM_CS0RCR1,’h0b010000);
    WR32(‘WEIM_CS0RCR2,’h00000000);
    WR32(‘WEIM_CS0WCR1,’h0c092480);
// Muxed memory
    WR32(‘WEIM_CS0GCR1,’h00410089);
    WR32(‘WEIM_CS0RCR1,’h0b010000);
    WR32(‘WEIM_CS0RCR2,’h00000000);
    WR32(‘WEIM_CS0WCR1,’h0c092480);
```

63.7.5.3 Samsung OneNand Synchronous Mode Configuration

Set memory and WEIM to synchronous read mode as shown:

```
WR16(‘CS0+(’hF221<<1),’hc0e0); // Synchronous read, 4 clk latency
WR32(‘WEIM_CS0GCR1,’h50412405); // 44 MHz (non-muxed)
WR32(‘WEIM_CS0RCR1,’h05010000);
```

The muxed Samsung OneNand supports synchronous write as shown:

```
// Set memory & WEIM to synchronous read and write mode
    WR16(‘CS0+(’hF221<<1),’hc0f2); // Sync. read and write, 4 clk latency
    WR32(‘WEIM_CS0GCR1,’h5041240f); // 44 MHz
    WR32(‘WEIM_CS0RCR1,’h05010000);
    WR32(‘WEIM_CS0WCR1,’h05040000);
```

63.7.5.4 Utility

The Samsung OneNand uses the following utility algorithms:

```
// Unlock Block command
    WR16(‘CS0+(’hF100<<1),’h0);        // DFS
    WR16(‘CS0+(’hF100<<1),’h0);        // DBS
    WR16(‘CS0+(’hF24c<<1),’h2);        // SBA - block number (2)
    WR16(‘CS0+(’hF241<<1),’h0);        // Clear interrupt status
    WR16(‘CS0+(’hF220<<1),’h23);       // Unlock command
    data =’h0;
    while(!(data &’h0004))             // Polling
       RD32(‘WIAR, data); // Read status
// Erase block command
    WR16(‘CS0+(’hF100<<1),’h2);        // DFS and block ([8:0]) address
    WR16(‘CS0+(’hF101<<1),’h0);        // DBS
    WR16(‘CS0+(’hF241<<1),’h0);        // Clear interrupt status
           WR16(‘CS0+(’hF220<<1),’h94);            // Erase command
```

```
        data ='h0;
        while(!(data &'h0004))                  // Wait
            RD32('WIAR, data); // Read status
    // Program page command
        WR16('CS0+('hF100<<1),'h2);             // DFS and block[8:0] address
        WR16('CS0+('hF107<<1),'h0);             // sector[1:0] and page[7:2] addresses
             WR16('CS0+('hF200<<1),'h800);          // buffer[11:8] address and counter[1:0]
        WR16('CS0+('hF241<<1),'h0);             // Clear interrupt status
        WR16('CS0+('hF220<<1),'h80);            // Program command
        data ='h0;
        while(!(data &'h0004))                  // Wait
            RD32('WIAR, data); // Read status
```

63.7.6 Access to Samsung UtRAM

The configurations in this section are intended for k1b5616b2m.

63.7.6.1 Samsung UtRAM Asynchronous Mode Configuration

Use the following for Samsung UtRAM asynchronous mode configuration.

```
        WR32('WEIM_CS0GCR1,'h400104b1);
        WR32('WEIM_CS0RCR1,'h0a010000);
        WR32('WEIM_CS0RCR2,'h00000008);
        WR32('WEIM_CS0WCR1,'h0b040040);
```

63.7.6.2 Samsung UtRAM Synchronous Mode Configuration

Use the following for Samsung UtRAM synchronous mode configuration.

```
RD16('CS0+('hff_ffff<<1),data);             // command sequence
        RD16('CS0+('hff_ffff<<1),data);
        RD16('CS0+('hff_ffff<<1),data);
        RD16('CS0+('hff_feff<<1),data);
        RD16('CS0+('h00_82a0<<1),data);             // memory sync. configuration
        WR32('WEIM_CS0GCR1,'h4021_53b7);            // fixed latency memory wrap 32
        WR32('WEIM_CS0RCR1,'h0500_0000);
        WR32('WEIM_CS0WCR1,'h0300_0000);
```

63.7.7 Access to Spansion Flash

The configurations in this section are for s29ws256n0s.

63.7.7.1 Boot

To boot from Spansion flash, use `BOOT_CFG= 'h261`.

63.7.7.2 Spansion Asynchronous Mode Configuration

Use the following for Spansion asynchronous mode configuration.

```
WR32('WEIM_CS0GCR1,'h00410081);
        WR32('WEIM_CS0RCR1,'h0a018000);
        WR32('WEIM_CS0RCR2,'h00000000);
```

```
          WR32('WEIM_CS0WCR1,'h0704a240);
WR16('CS0+('hF220<<1),'h94);            // Erase command
          data ='h0;
          while(!(data &'h0004))                  // Wait
              RD32('WIAR, data); // Read status
      // Program page command
          WR16('CS0+('hF100<<1),'h2);             // DFS and block[8:0] address
          WR16('CS0+('hF107<<1),'h0);             // sector[1:0] and page[7:2] addresses
          WR16('CS0+('hF200<<1),'h800);           // buffer[11:8] address and counter[1:0]
          WR16('CS0+('hF241<<1),'h0);             // Clear interrupt status
          WR16('CS0+('hF220<<1),'h80);            // Program command
          data ='h0;
          while(!(data &'h0004))                  // Wait
              RD32('WIAR, data); // Read status
```

63.7.7.3 Spansion Synchronous Mode Configuration

Use the following for Spansion synchronous mode configuration.

```
WR16('CS0+('h0555<<1),'h00aa);   // command sequence
          WR16('CS0+('h02aa<<1),'h0055);
          WR16('CS0+('h0555<<1),'hd0);
          WR16('CS0+('h0000<<1),'h1ec4);   // memory sync. configuration
          WR32('WEIM_CS0GCR1,'h50411325); // 66 MHz
          WR32('WEIM_CS0RCR1,'h05000000); // 5 cycles on memory
```

63.7.7.4 Utility

The Spansion Flash device uses the following utility algorithms:

```
      // Single word programming
          COMMAND_SEQUENCE(cs,16,'ha0);           // single word programming
          WR16('CS0+addr, data);
          CHECK_STATUS('CS0+addr,data,16,1,errst);
      // Write buffer programming
          COMMAND_SEQUENCE(0,16,'h25);              // write buffer programming
          WR16('CS0+addr,'h001f);                     // counter-1
          WR_I('CS0+addr, data, 'h0010_0001, 16); // data
          WR16('CS0+addr,'h0029);                     // write buffer to flash
          CHECK_STATUS('CS0+addr+'h3e,data[31:16]+'h00f0,16,1,errst);
There COMMAND_SEQUENCE and CHECK_STATUS are next functions:
      task COMMAND_SEQUENCE;
          input[2:0]    cs;
          input[7:0]    port_size;
          input[31:0]   code;
      begin
          if(port_size == 16)
             begin
             WR16(csba[cs]+('h0555<<1),'h00aa);
             WR16(csba[cs]+('h02aa<<1),'h0055);
             WR16(csba[cs]+('h0555<<1),code);
             end
          else
             begin
             WR32(csba[cs]+('h0555<<2),'h00aa);
             WR32(csba[cs]+('h02aa<<2),'h0055);
             WR32(csba[cs]+('h0555<<2),code);
```

```
            end
        end
        endtask
        task     CHECK_STATUS;
            input[31:0] addr;
            input[31:0] edata;
            input[7:0]   port_size;
            input[7:0]   opcode;
            output[7:0] errst;
            reg[31:0]    data;
            reg[31:0]    data3;
        begin
            errst = 0;
            data  = 0;
            data3 = 0;
while(!(data == edata) && !errst) // Wait operation
          begin: BR_EN
          RD16(addr, data);             // Read status
          if(data[7] !== edata[7])
              begin
              if(data[5] == 1)
                   begin
                   RD16(addr, data3);
                   RD16(addr, data);
                   if(data[6] !== data3[6])
                        begin
                        $display("CHECK_STATUS: Error timout on single data program");
                        errst = 1;
                        disable BR_EN;
                        end
                   end
              else
                   begin
                   if(opcode == 2)
                        if(data[1] == 1)
                            begin
                            RD16(addr, data3);
                            if(port_size == 32)
                               RD32(addr, data);
                            else
                               RD16(addr, data);
                            if(data[1] == 1 && data !== edata)
                               begin
                               $display("CHECK_STATUS: Error on write buffer");
                               errst =3;
                               disable BR_EN;
                               end
                            end
                   end
              end
          else
              begin
              RD16(addr, data3);
              if(port_size == 32)
                   RD32(addr, data);
              else
                   begin
```

```
                RD16(addr, data);
                edata[31:16] = 16'h0;
                end
            if(data !== edata)
                begin
                $display("CHECK_STATUS: Error in data write on single data program");
                errst =2;
                disable BR_EN;
                end
            end
        end
end
endtask
```

63.7.8 8-Bit Support

For intel mode, use the connections as shown in Table 63-24.

Table 63-24. Intel Mode Pin Connections

CPU Pin	WEIM Pin	Notes
ADS#	IPP_DO_ADV_B	WAL = 1,RAL = 1
W/R	IPP_DO_BE_B	WBED = 1
WR#	WE#	—
RD#	OE#	—

For Motorola mode, use the connections as shown in Table 63-25:

Table 63-25. Motorola Mode Pin Connections

CPU Pin	WEIM Pin	Notes
AS#	IPP_DO_CS_B	—
R/W#	WE#	—
LDS#	BE#	—

63.8 Booting from One NAND Devices

The following timing diagrams show the timing of accesses to memory or a peripheral with different timing parameters. All examples are specifically for CS0, but they are valid for any other chip select. BE means one from current used BE[3:0].

63.8.1 Asynchronous Read Memory Accesses Timing Diagram

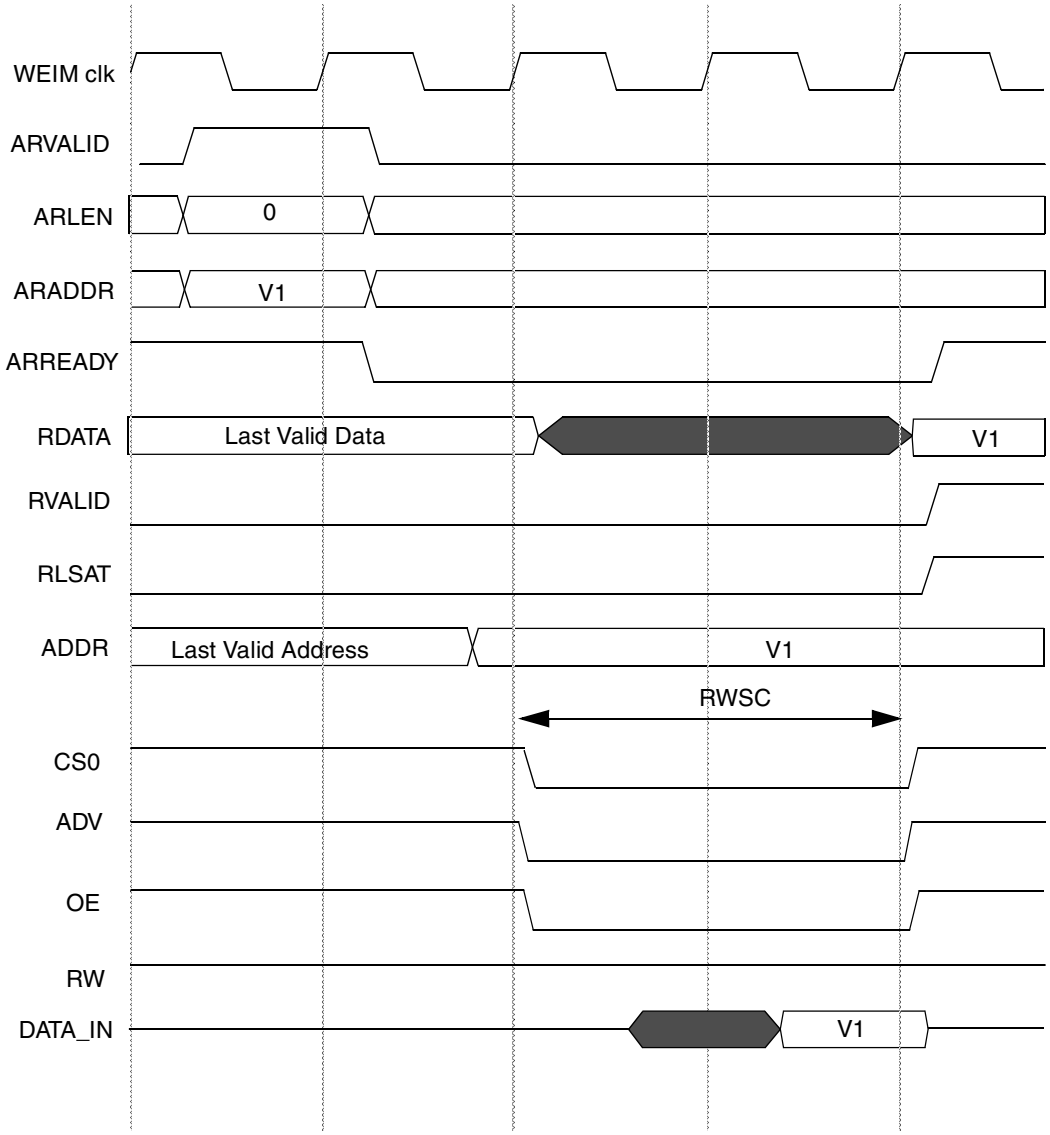


Figure 63-12. Read Access, RWSC=2,RCSA=0,OEA=0,RCSN=0,OEN=0, RAL=1

63.8.2 Asynchronous Write Memory Accesses Timing Diagram

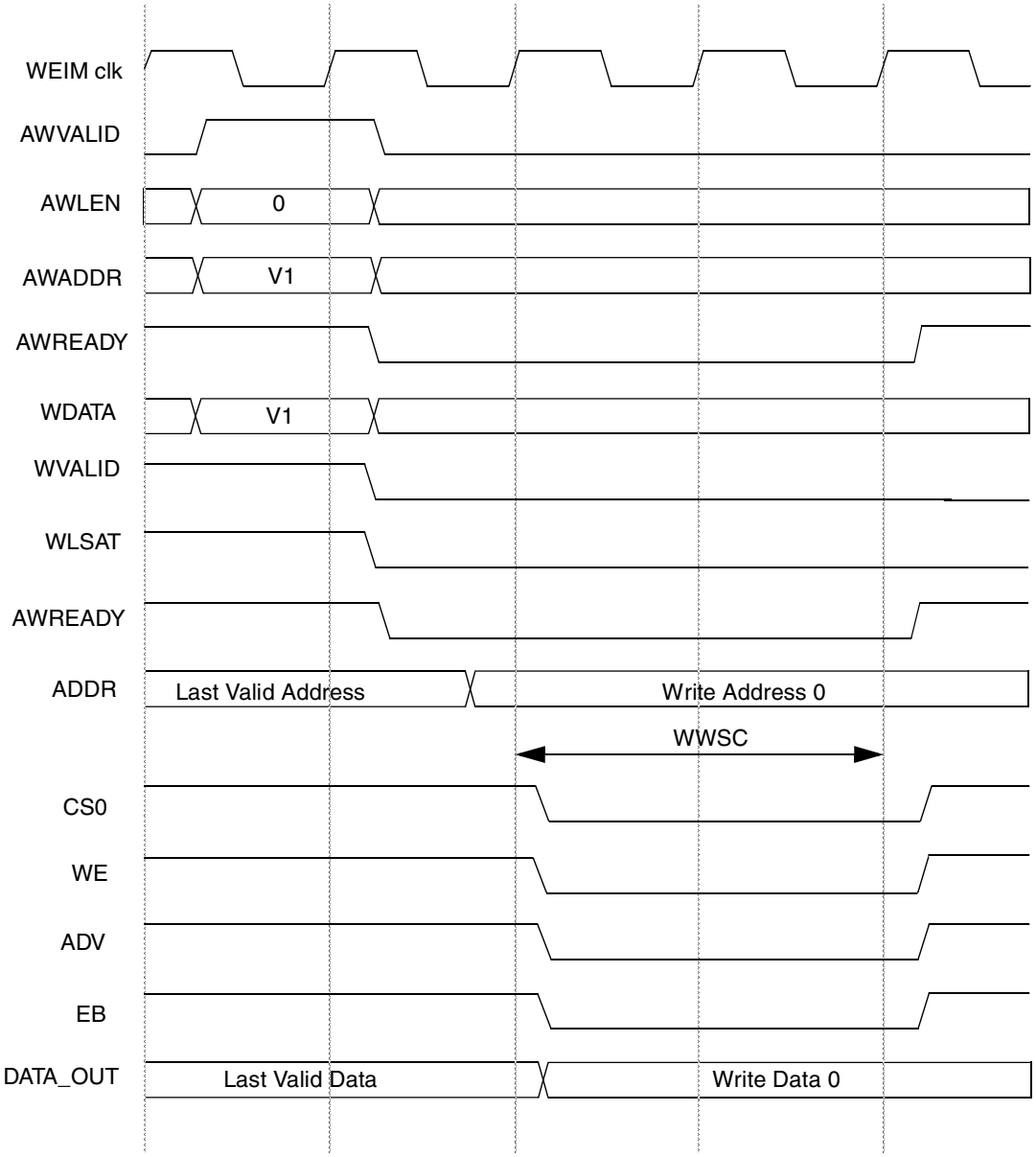


Figure 63-13. Write Access, WWSC=2,WCSA=0,WEA=0,WCSN=0,WEN=0,BEA=0, BEN=0, WAL=1

63.8.3 Asynchronous Read/Write Memory Accesses Timing Diagram

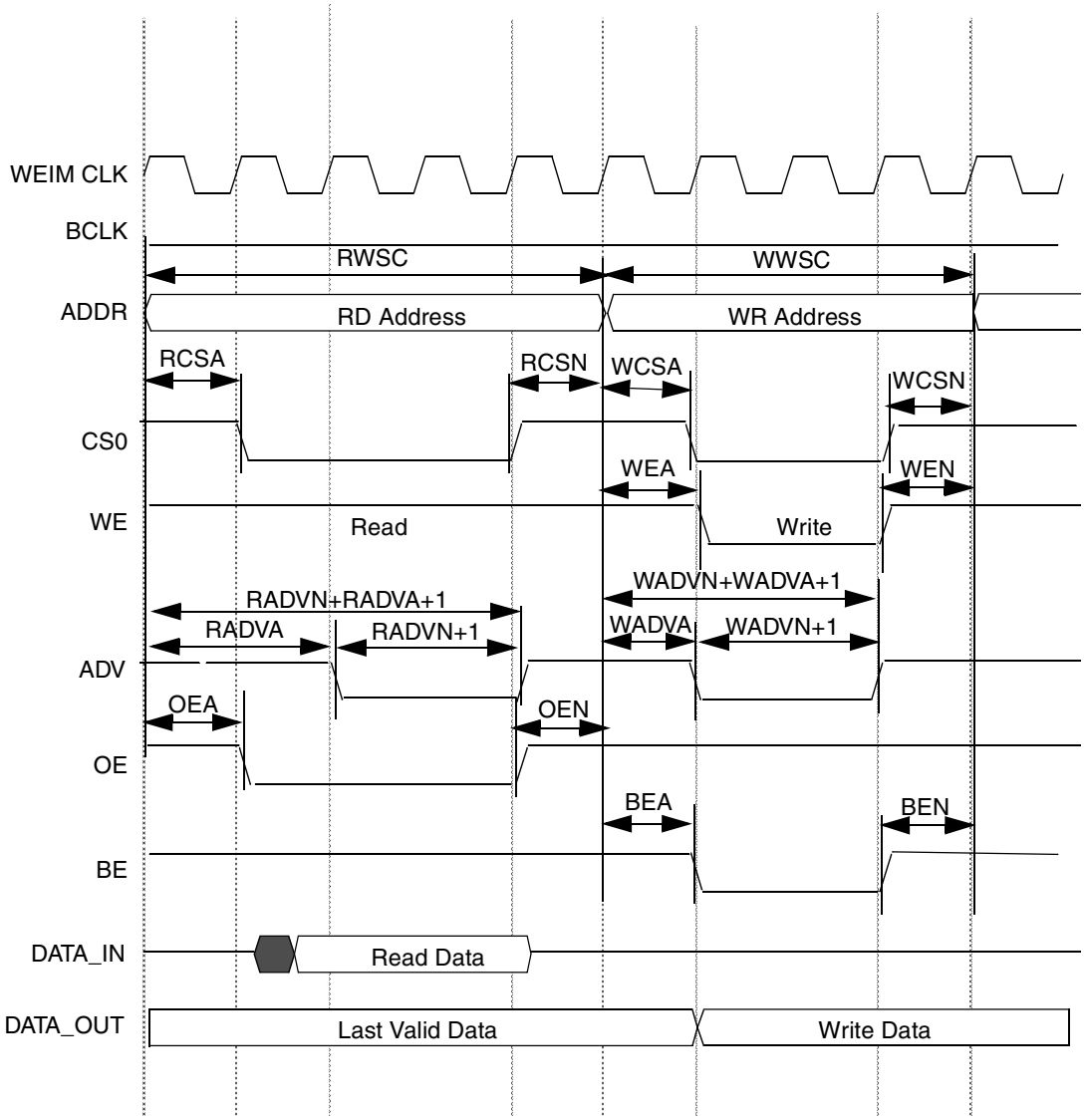


Figure 63-14. RCSA=1,RADVA=2,OEA=1,RADVN=1,RCSN=1,OEN=1,WCSA=1,WEA=1,WADVA=1,BEA=1,WADVN=1,WCSN=1,WEN=1,BEN=1

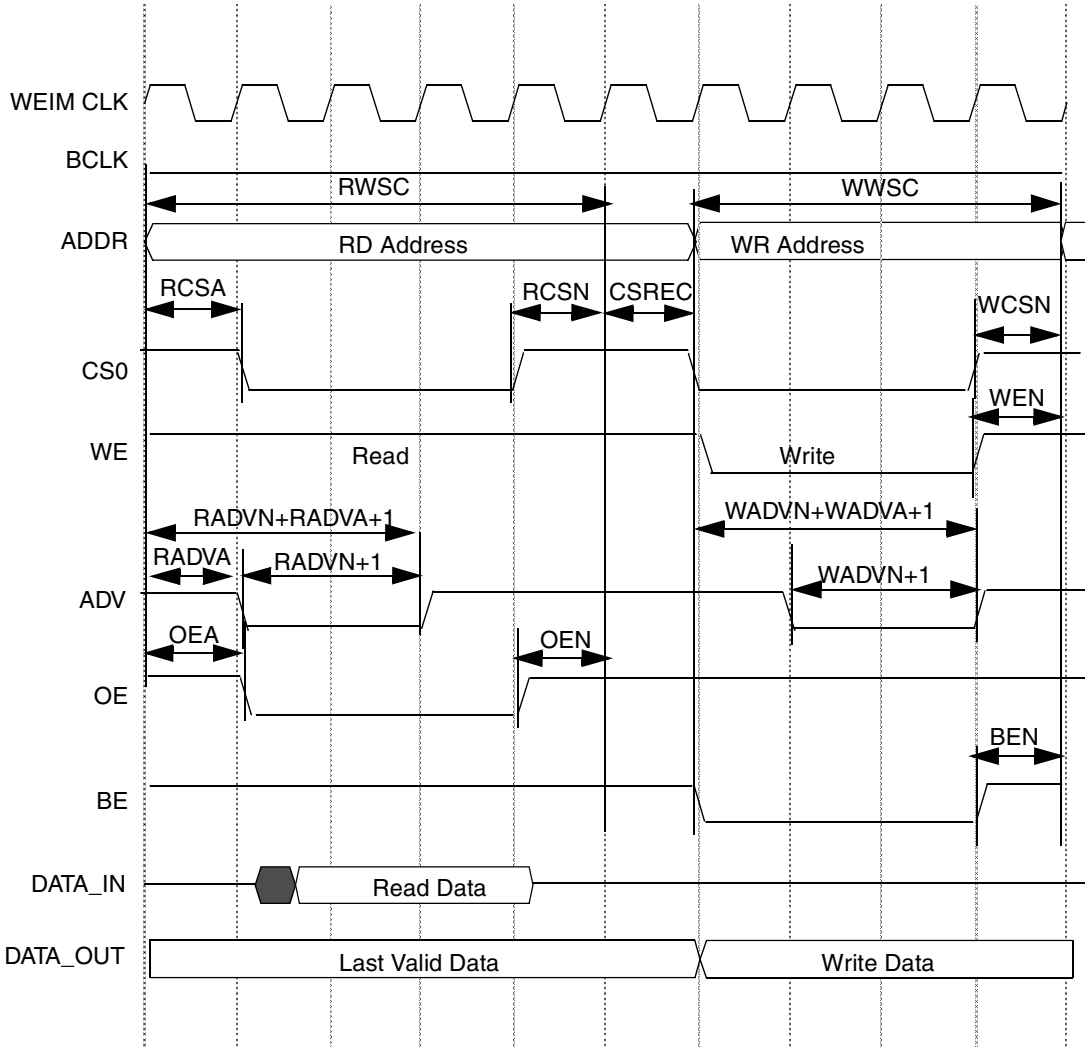


Figure 63-15. RWSC=5,RCSA=1,RCSN=1,RADVA=1,RADVN=1,OEA=1,OEN=1,WWSC=4,WCSA=0,WCSN=1, WEA=0,WEN=1,WADVA=1,WADVN=1,BEA=0,BEN=1,CSREC=1

63.8.4 Asynchronous Read/Write using RAL, WAL and CSREC

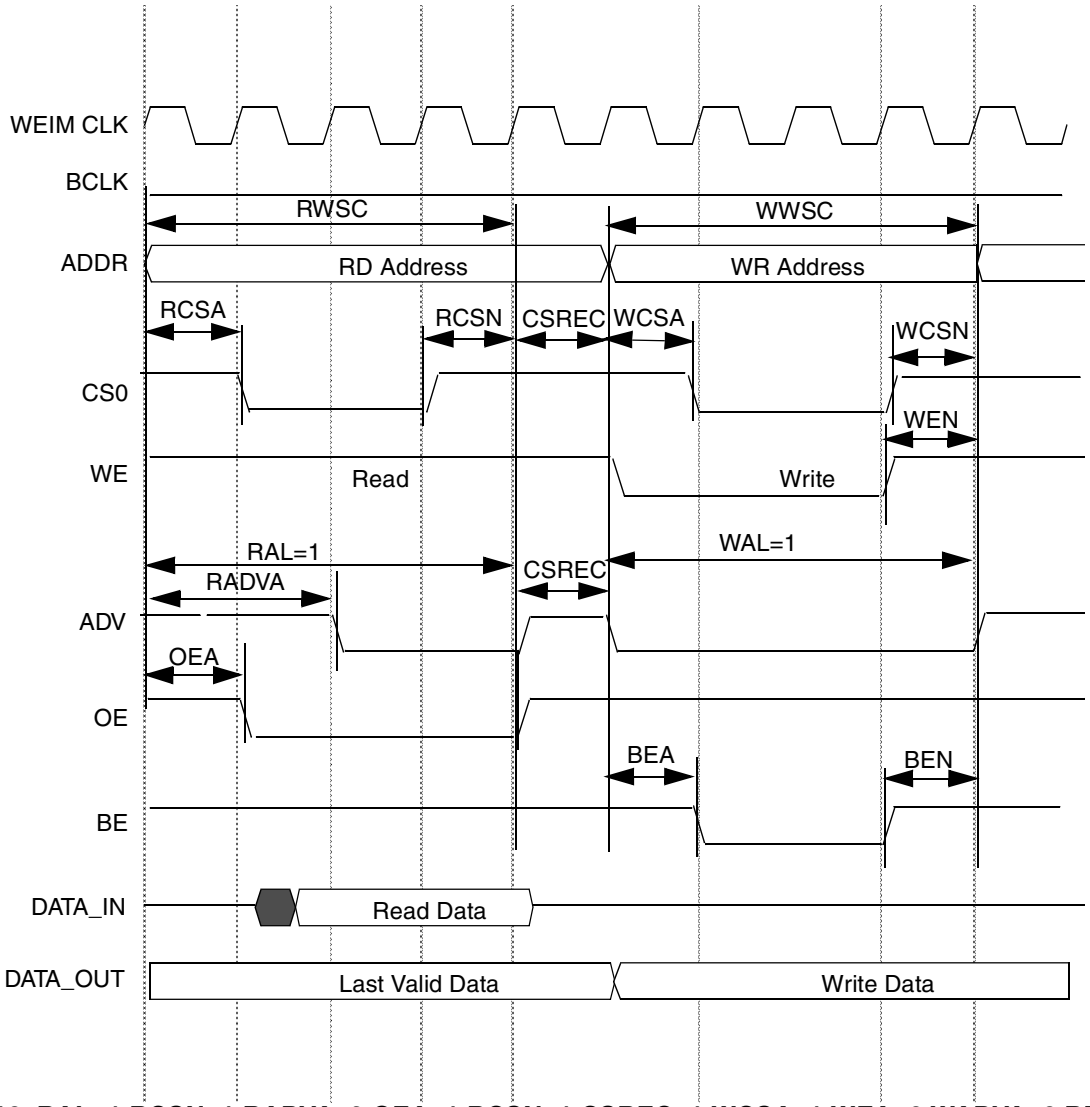


Figure 63-16. RAL=1,RCSN=1,RADVA=2,OEA=1,RCSN=1,CSREC=1,WCSA=1,WEA=0,WADVA=0,BEA=1,WAL=1,WCSN=1,WEN=1,BEN=1

63.8.5 Consecutive Asynchronous Write Memory Accesses Timing Diagram

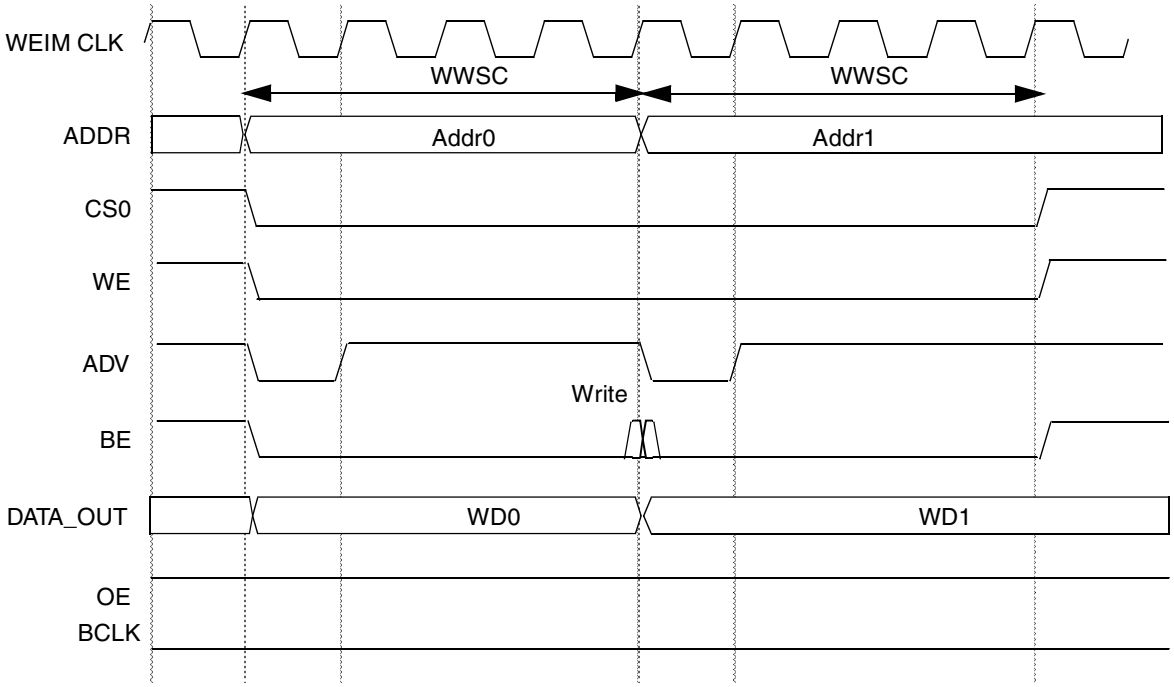


Figure 63-17. WWSC=4,WCSA=0,WEA=0,WADVA=0,BEA=0,WCSN=0,WEN=0,WADVN=0,BEN=0,CSREC=0

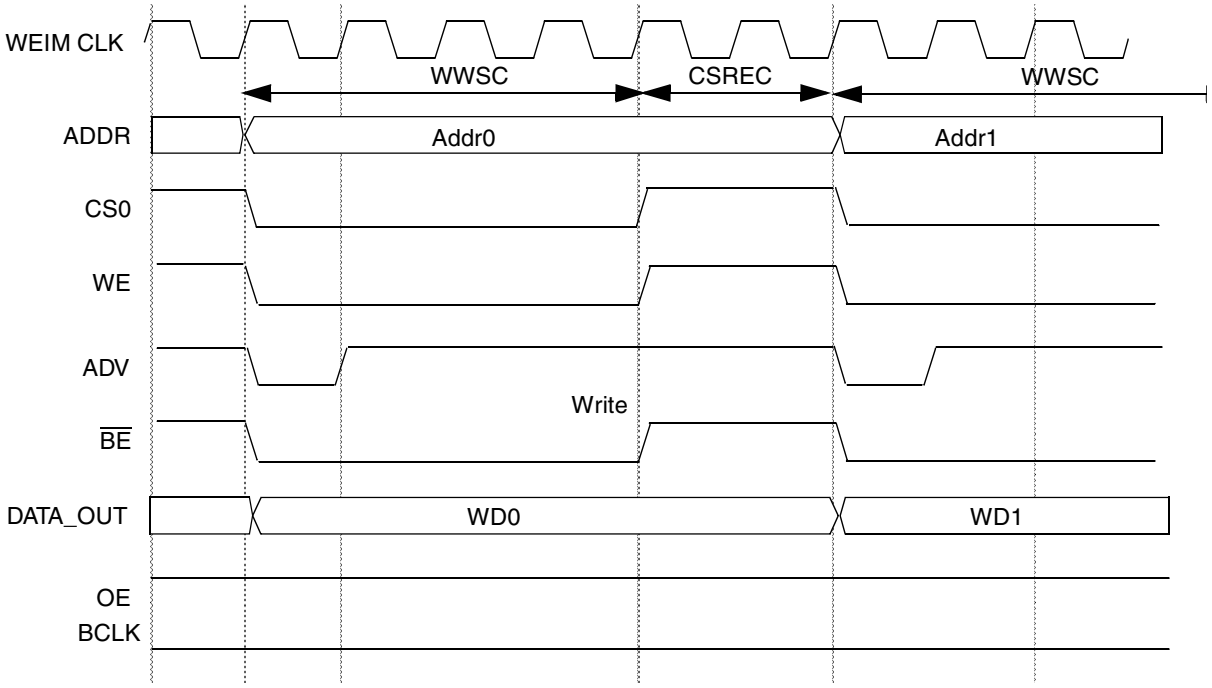


Figure 63-18. WWSC=4,WCSA=0,WEA=0,WADVA=0,BEA=0,WCSN=0,WEN=0,WADVN=0,BEN=0,CSREC=2

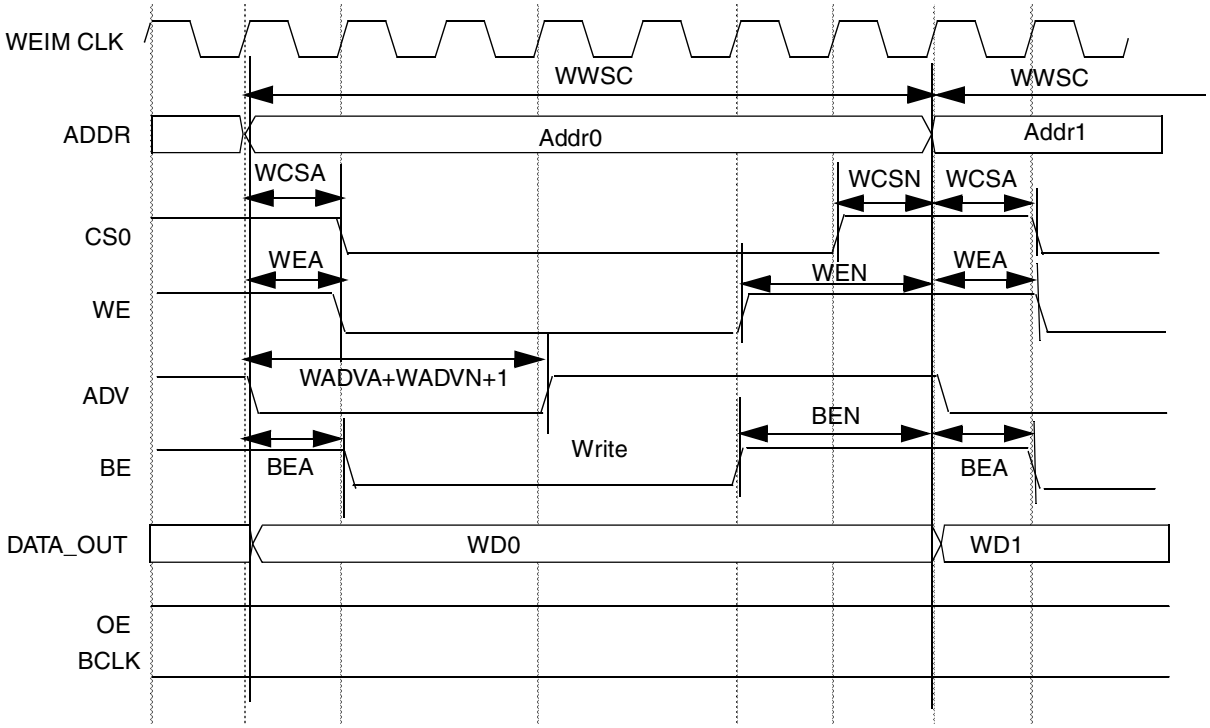


Figure 63-19. WWSC=7,WCSA=1,WCSN=1,WEA=1,WEN=2,WADVA=0,WADVN=2,BEA=1,BEN=2

63.8.6 Consecutive Asynchronous Read Memory Accesses Timing Diagram

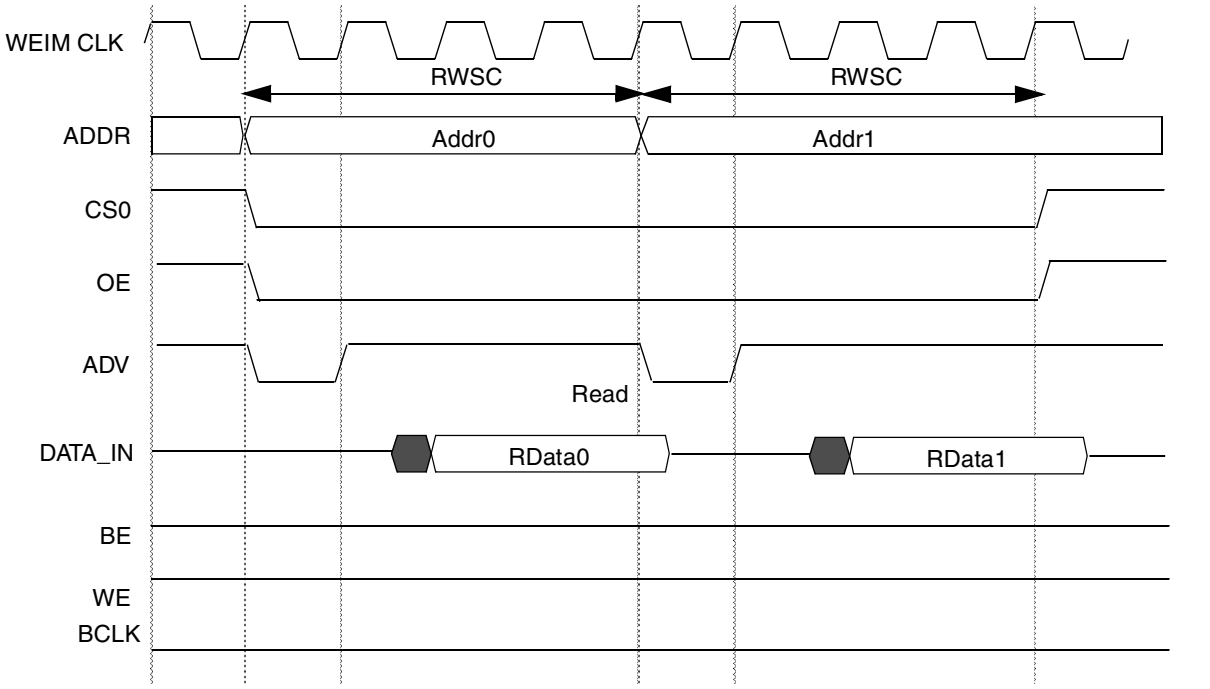


Figure 63-20. RWSC=4,RCSA=0,OEA=0,RADVA=0,RCSN=0,OEN=0,RADVN=0,CSREC=0

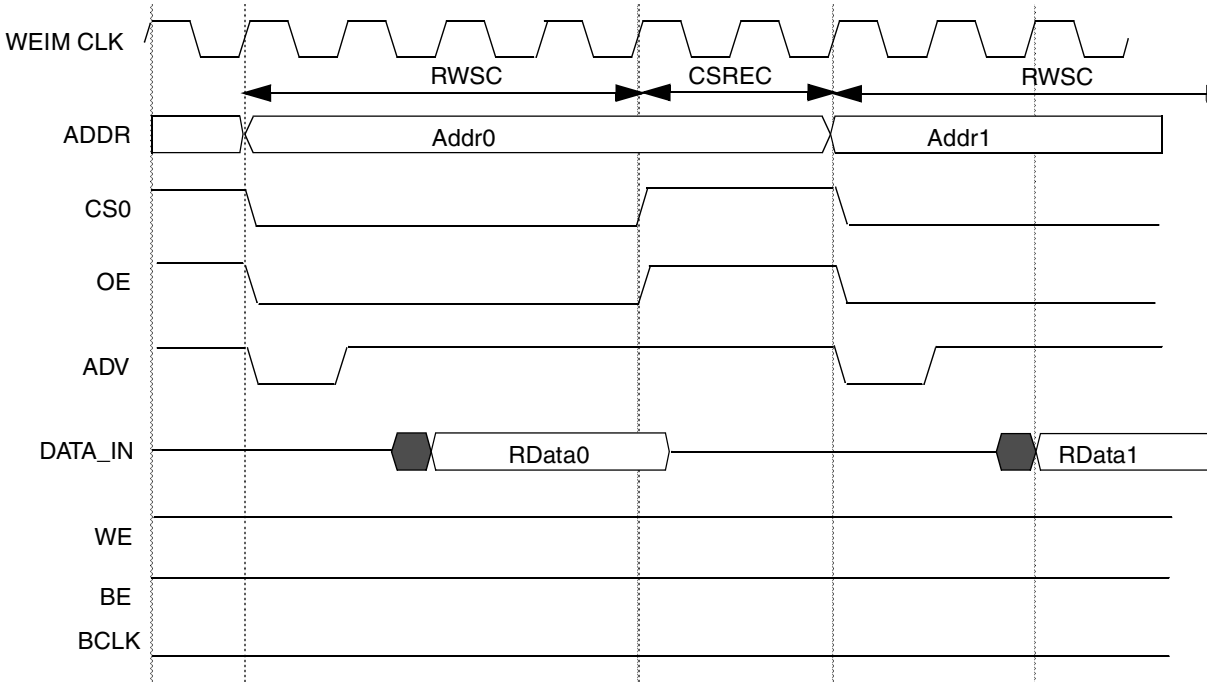


Figure 63-21. RWSC=4,RCSA=0,OEA=0,RADVA=0,RCSN=0,OEN=0,RADVN=0,CSREC=2

63.8.7 Burst Read Memory Accesses Timing Diagram - BCD=0

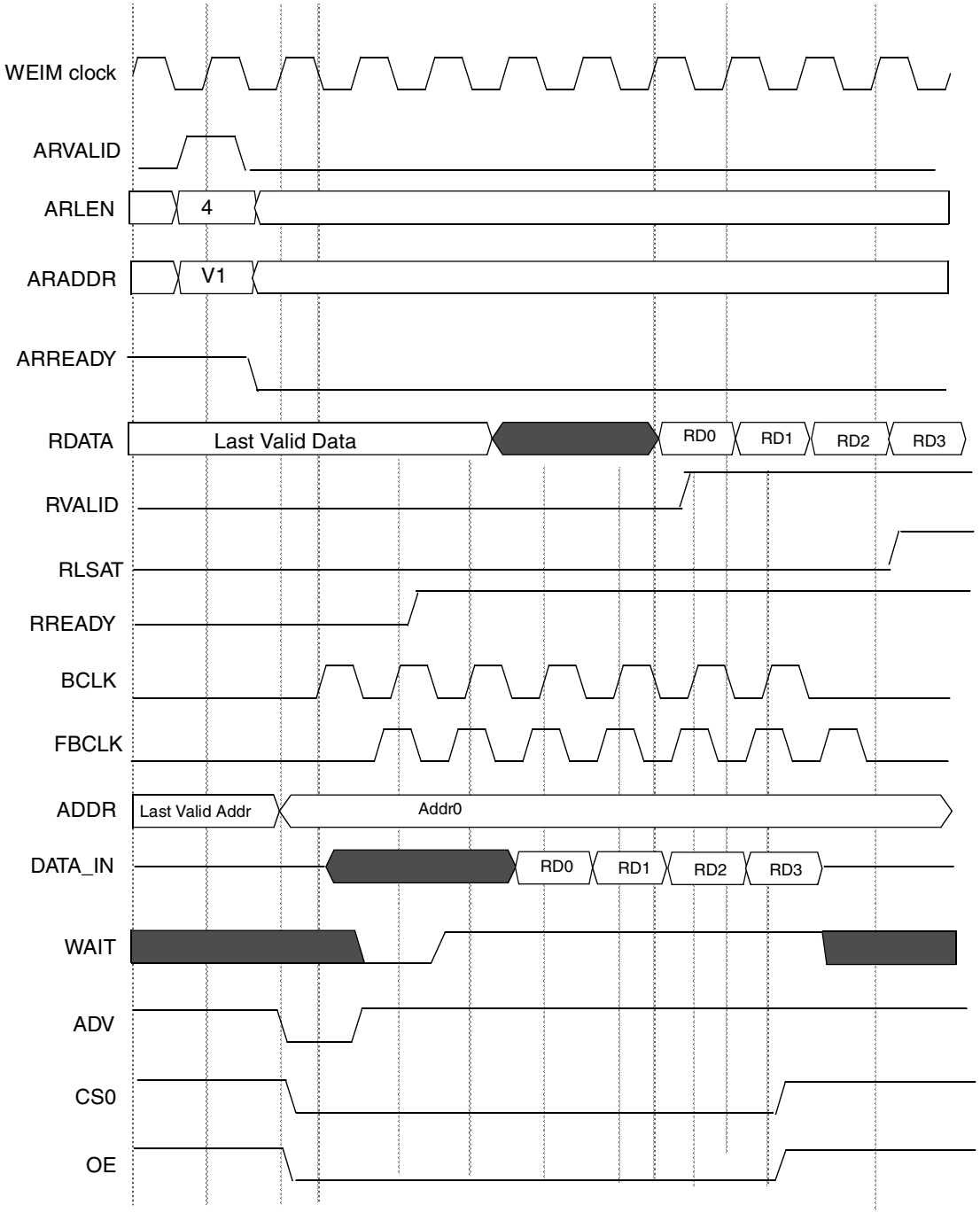


Figure 63-22. SRD=1,BCD=0,BCS=0,RWSC=1,RADVA=0,RADVN=0,RFL=0,RL=0

63.8.8 Burst Read Memory Accesses Timing Diagram - BCD=1

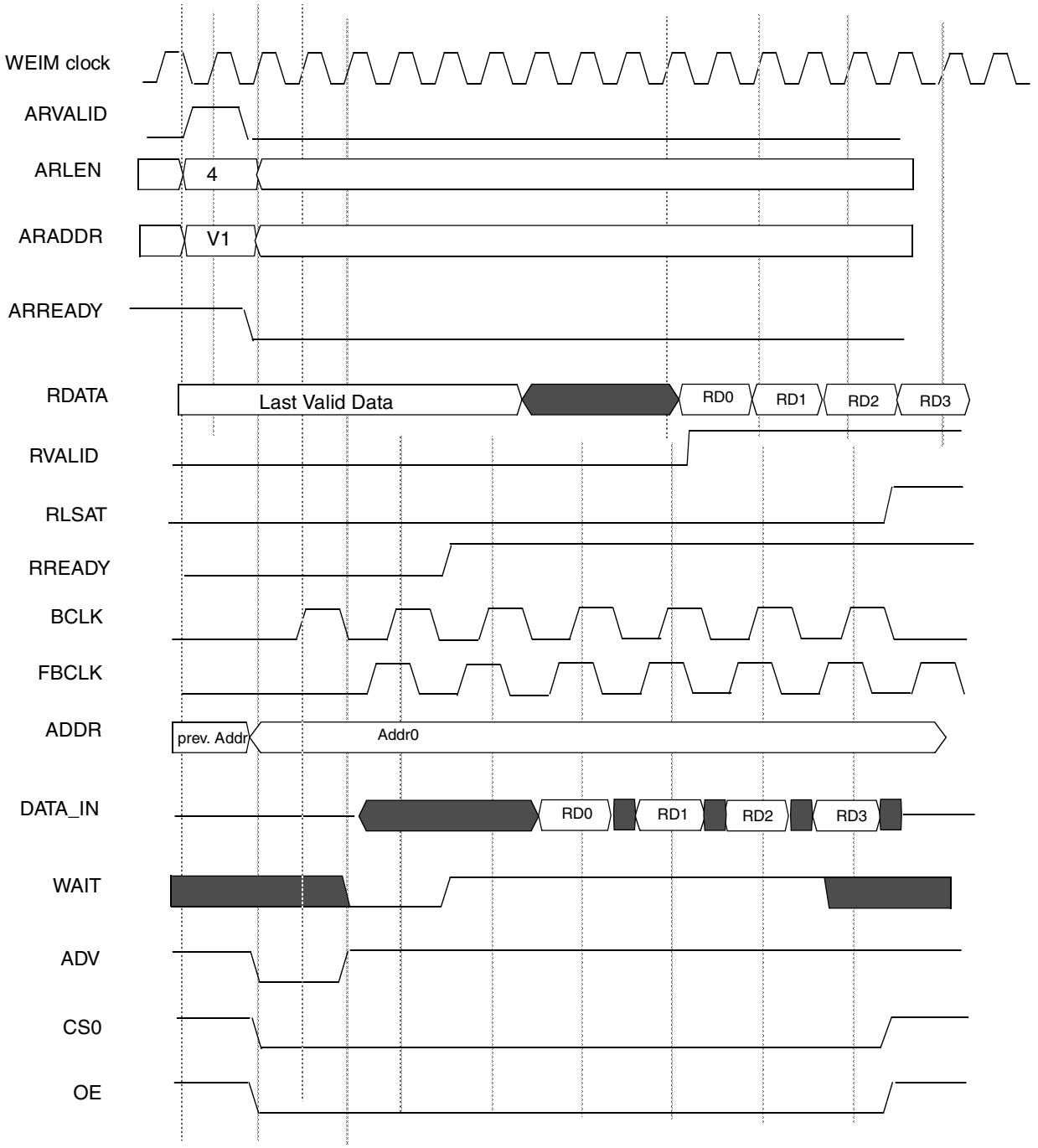


Figure 63-23. BCD=1, RL = 3

63.8.9 Async. Page Mode Access

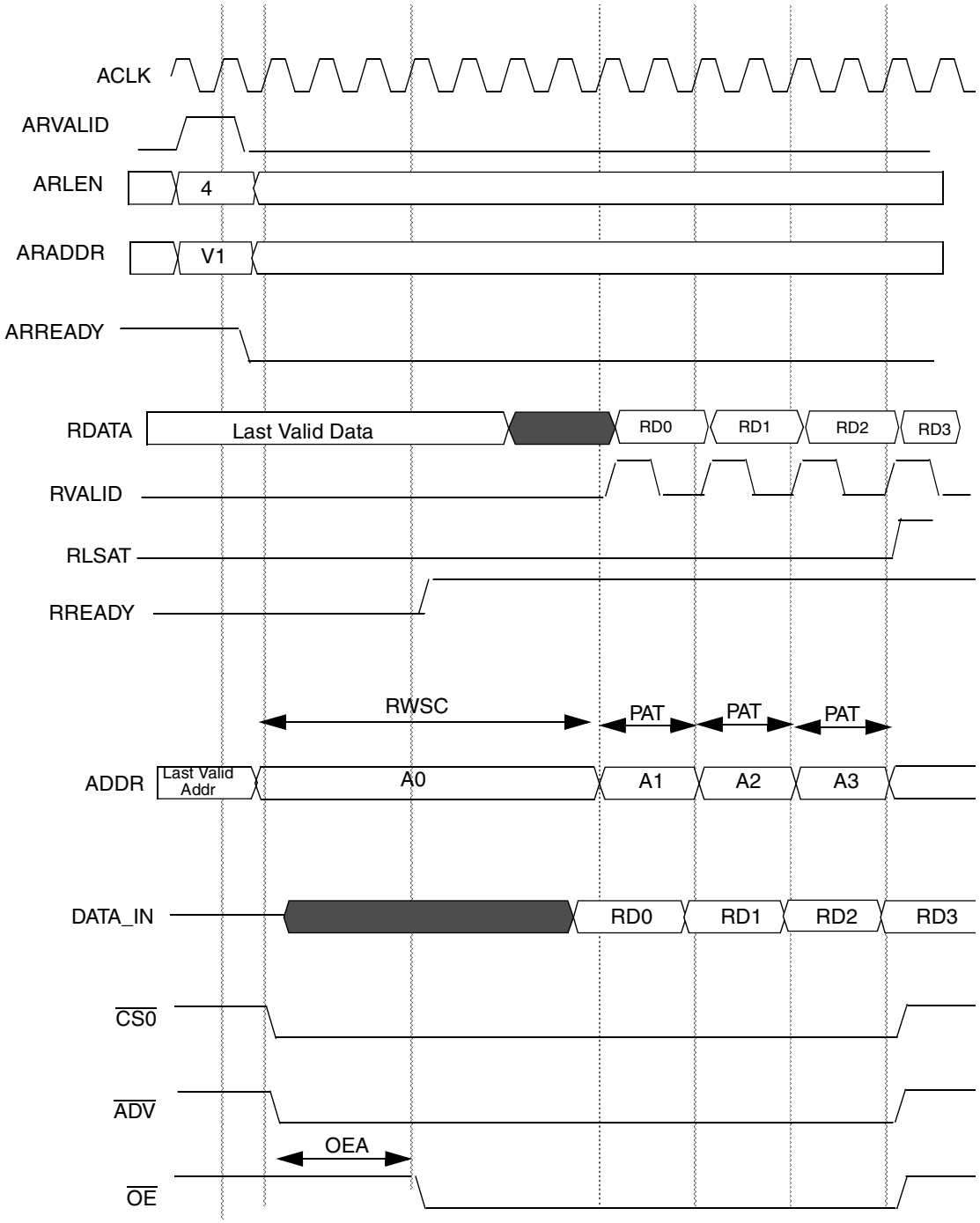


Figure 63-24. PAT = 2

63.8.10 DTACK Mode—AXI Single Access

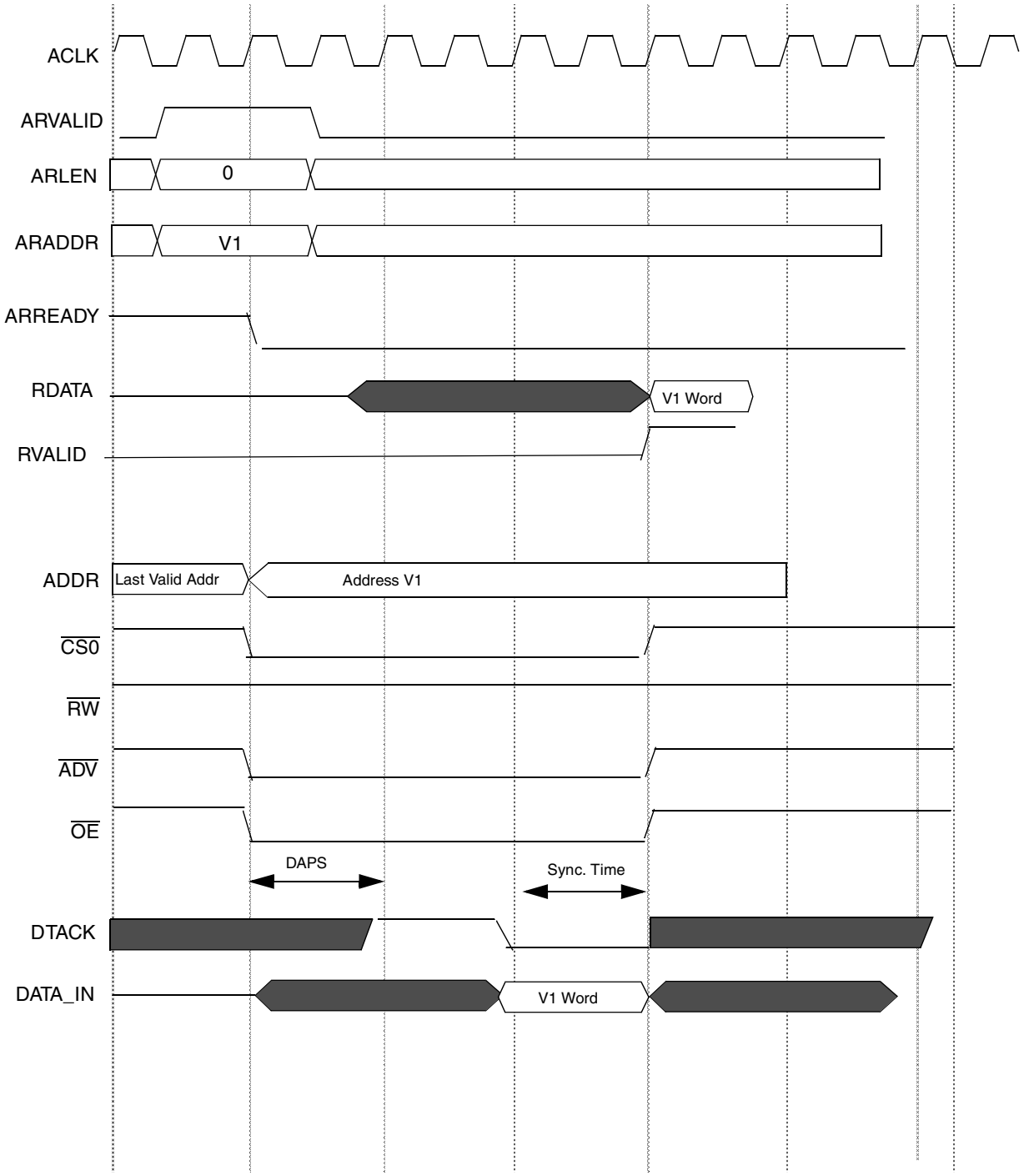


Figure 63-25. DAPS = 2

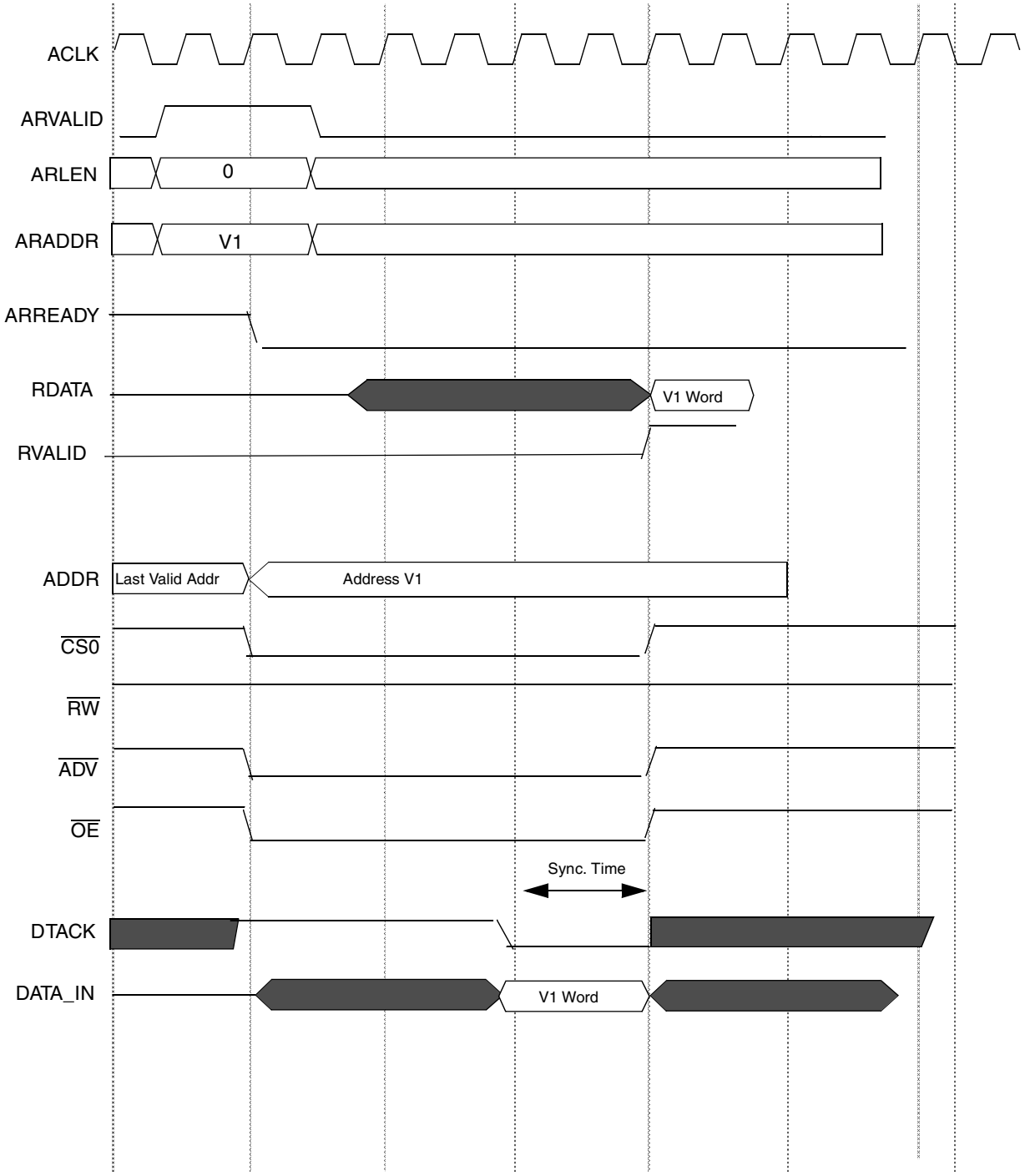


Figure 63-26. DAPS = 0

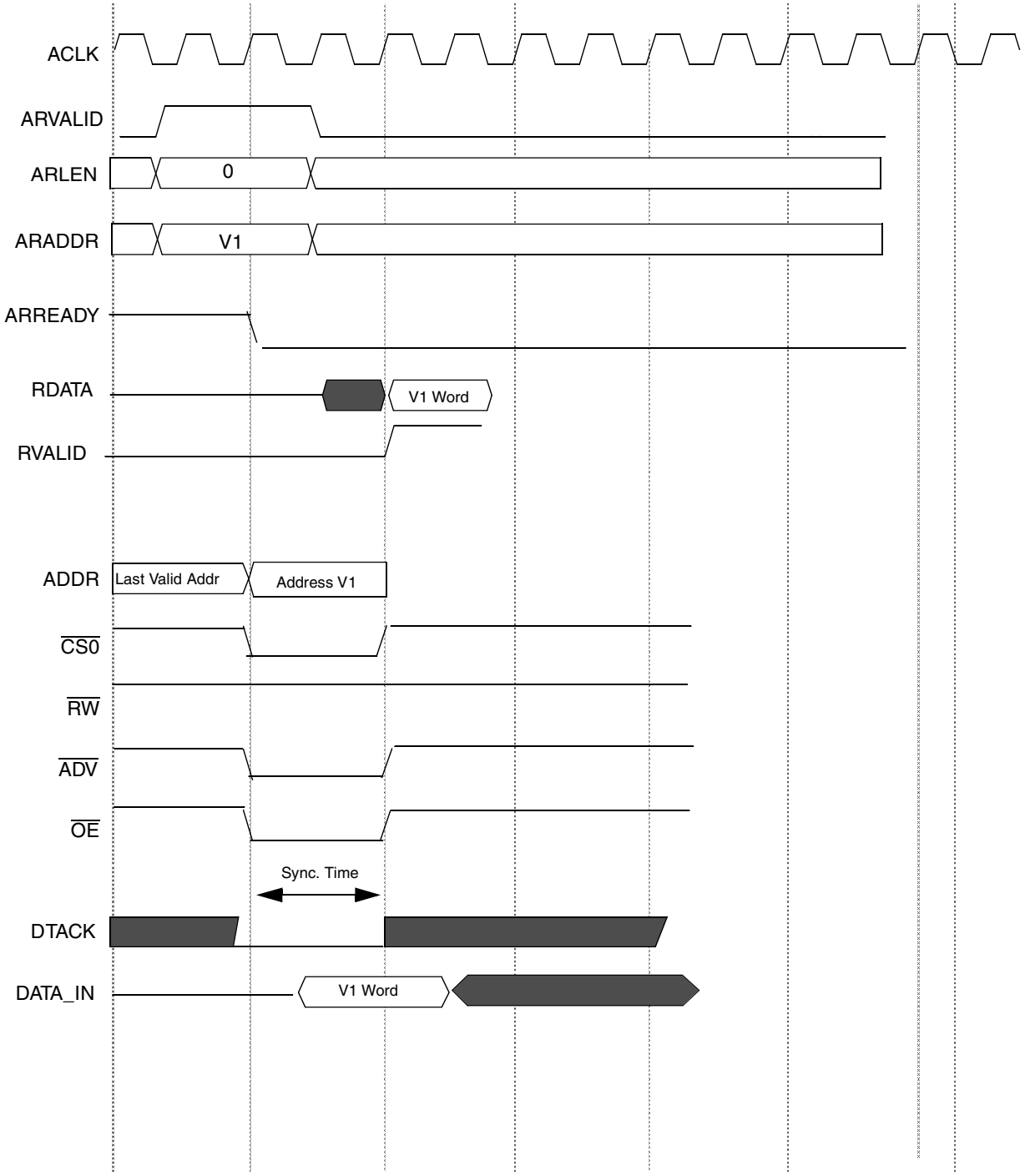


Figure 63-27. DAPS = 0

63.8.11 DTACK Mode—AXI Burst Access

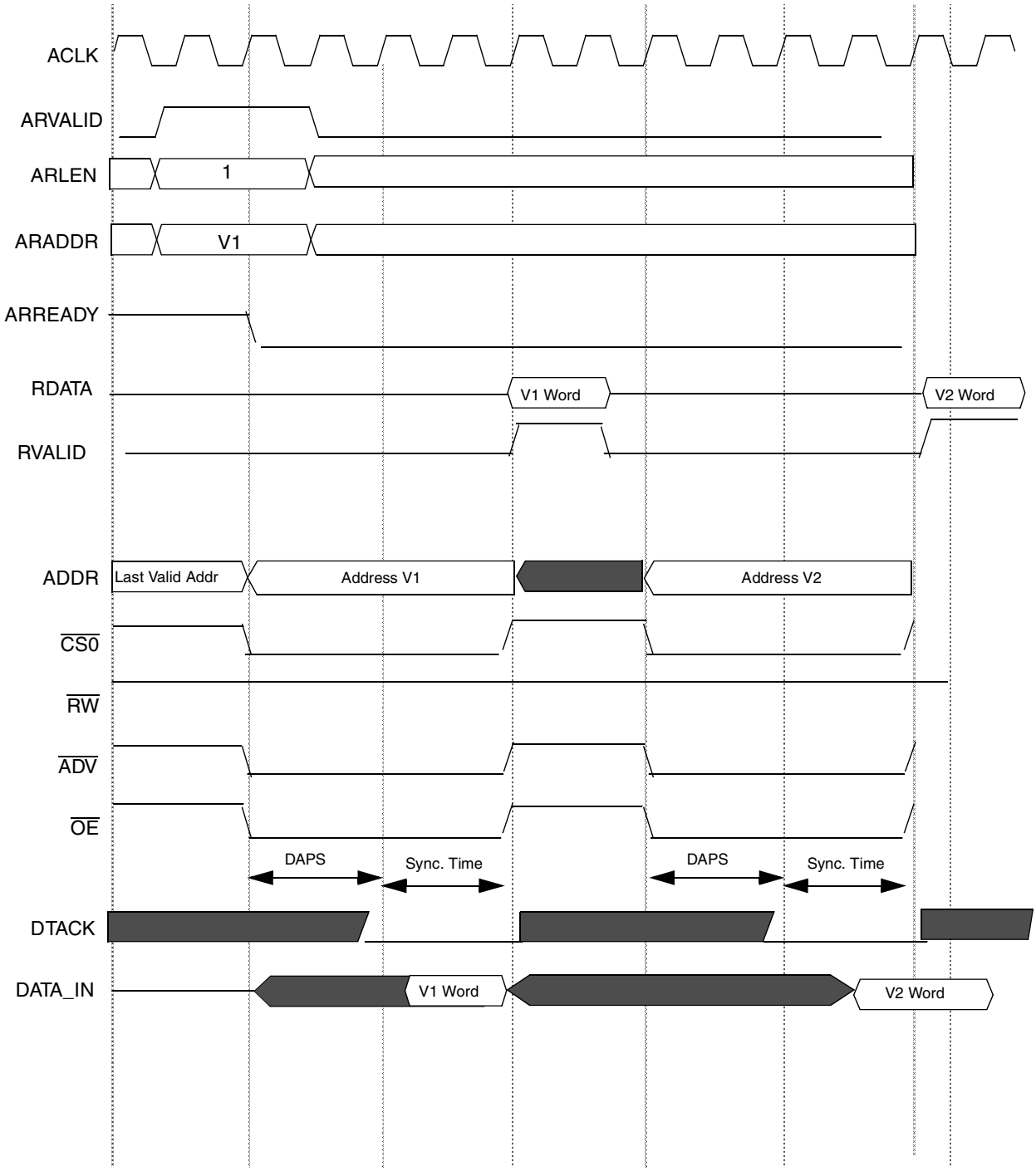


Figure 63-28. DAPS = 2 CSREC = 2

Appendix A
IOMUX Controller (IOMUXC)

A.1 Introduction

A.1.1 Overview

The IOMUX controller (IOMUXC), together with the IOMUX, enables the i.MX51 to share one BGA contact with several functional blocks. The sharing is done by multiplexing the BGA contact input/output signals. For each BGA contact there are up to 8 muxing options (called ALT modes). Because different modules require different BGA contact settings (like pull up, keeper, etc), the IOMUXC also controls the BGA contact settings parameters.

The IOMUX consist of only combinatorial logic combined from several basic iomux cells, each basic iomux cell handles only one BGA contact signals muxing.

Figure A-1 illustrates the IOMUX/IOMUXC connectivity in the system.

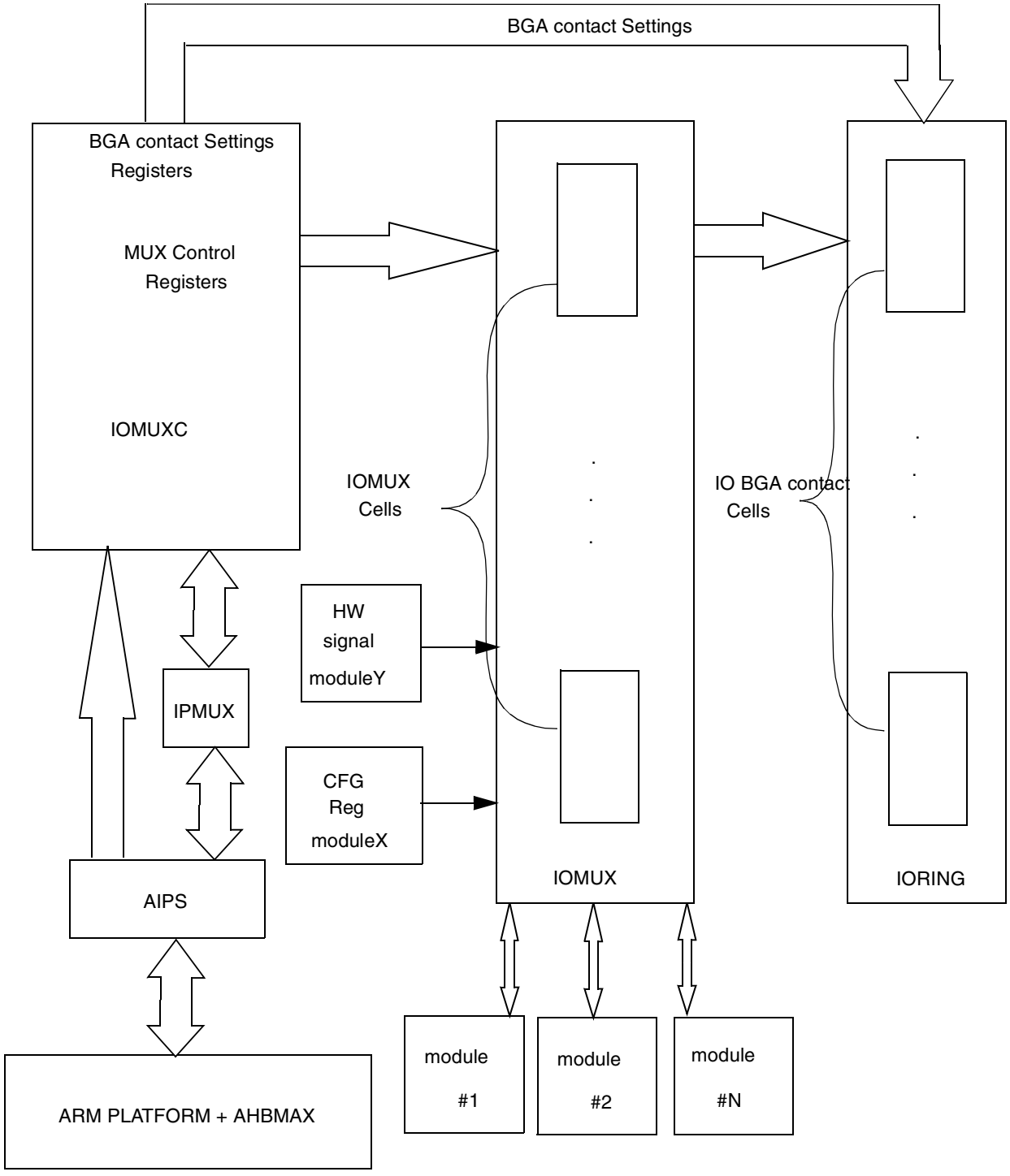


Figure A-1. IOMUX SoC Level Block Diagram

A.1.2 Features

The IOMUXC features are as follows:

- 32 bits SW Mux control registers (IOMUXC_SW_MUX_CTL_PAD_<BGA contact NAME> or IOMUXC_SW_MUX_CTL_GRP_<GROUP NAME>) to configure a specific mux mode of each BGA contact or a predefined group of pads and to enable forcing the input path of the BGA contact/s (SION bit).
- 32 bits SW BGA contact control registers (IOMUXC_SW_PAD_CTL_PAD_<PAD_NAME> or IOMUXC_SW_PAD_CTL_GRP_<GROUP NAME>) to configure specific BGA contact settings of each contact or a predefined group of pads.
- 32 bits General Purpose Registers—Two 32 bits registers according to SOC requirements for any usage.
- 32 bits “Daisy Chain” Control registers—register to control the input path to a module when more than one contact may drive this module input.
- IPS Interface—Peripheral bus to allow registers configuration and readiness (all registers are read/write registers).

A.1.2.1 SW Mux and BGA contact Control

For the SW_PAD_CTL_PAD and SW_PAD_CTL_GRP registers, each I/O configuration parameter uses two different indicators as follows:

- Associated bits shown below the white bit-location boxes are the default values for that parameter. These values are configurable by software.
- Associated bits shown below the gray bit-location boxes (read-only) are always zero and are simply the values read from that particular register location and they do not reflect the actual configuration default. These values are not configurable by software. For default configuration parameter values, refer to Chapter 4, “External Signals and Pin Multiplexing.”

A.1.3 Modes of Operation

There is only one mode of operation: normal mode.

A.2 External Signal Description

There are no signals in the IOMUX that connect off chip (they all control the PADS functionality).

A.3 Functional Description

The IOMUXC Detailed Block Diagram is illustrated in Figure A-2.

The IOMUXC is consist of 2 sub blocks:

- IOMUXC_REGISTERS: includes all the IOMUXC registers (6 main types are illustrated).
- IOMUXC_LOGIC: includes all the IOMUXC combinatorial logic (IP interface controls, address decoder, observability muxes).

The IOMUXC external signals and the IOMUXC sub blocks connections internal signals are also detailed in this diagram. The following are the naming conventions used along with some explanation:

- <BGA contact> - any BGA contact logic name.
- <GRP> - any BGA contact group logic name.
- <FUNC> - one of the following BGA contact control functions:
  - SRE (1 bit slew rate control).
  - DSE (2 bits drive strength control).
  - ODE (1 bit open drain control).
  - HYS (1 bit hysteresis control).
  - PULL_KEEP_CTL (4 bits pull up/down and keeper controls)
  - PUS (2 bits pull up/down configuration value)
  - PUE (1 bit pull/keep select)
  - PKE (1 bit enable/disable pull up, pull down or keeper capability)
  - DDR_MODE_SEL (1 bit ddr_mode control)
  - DDR_INPUT (1 bit ddr_input control)
- ips_addr[X:2] - since the IOMUXC registers are all 32 bits width address lines 0,1 are not used. X will be calculated automatically by the tool according to the final register number: X is smallest integer value which satisfy the constraint: 2^(X-1) >= (Total number of IOMUXC registers).

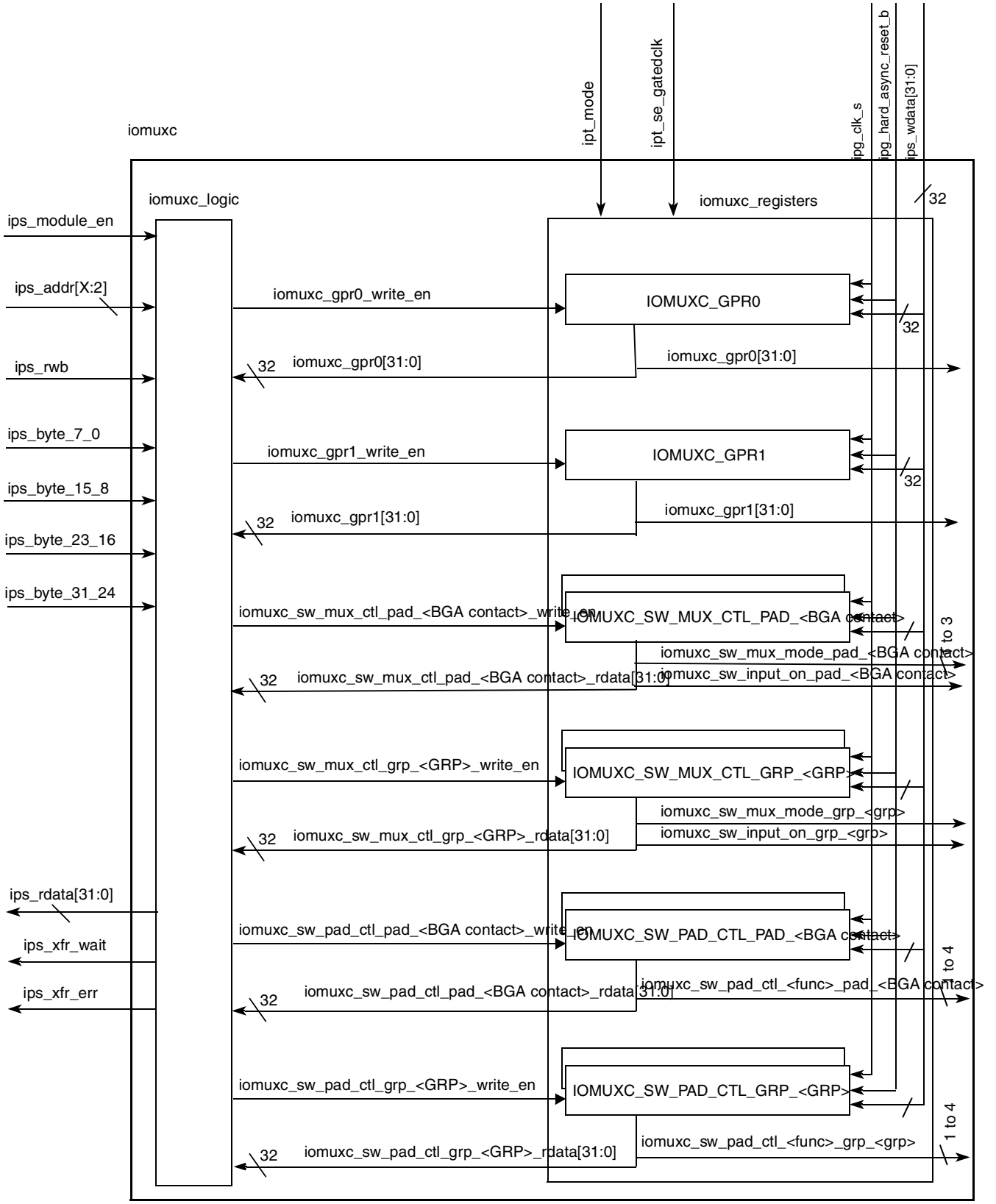


Figure A-2. IOMUXC Block Diagram

The IOMUX consist of a number (about the number of pads in the SoC) of basic iomux_cell units. If only one functional mode is required for a specific BGA contact then there is no need for IOMUXing and the signals can be connected directly from the module to the IORING. IOMUX cell is required whenever 2 or more functional modes are required for a specific BGA contact or when one functional mode and the one test mode are required.

The basic iomux_cell design, which allows 2 levels of HW signal control (in ALT6 and ALT7 modes - ALT7 gets highest priority) is shown in Figure A-3.

A.3.1 ALT6 and ALT7 Extended Muxing Modes

The ALT7 and ALT6 Extended Muxing Modes allows any signal in the system (Fuse, BGA contact input, JTAG, SW register, etc) to override any SW configuration and to force the ALT6/ALT7 Muxing Mode. It also allows an IOMUX SW register to control a group of pads.

A.3.2 SW Loopback through SION Bit

A limited but satisfying option exist to override the regular BGA contact functionality and force the input path to be active (ipp_ibe==1'b1) regardless of the value driven by the corresponding module. This can be done by setting the SION (SW Input On) bit in the IOMUXC_SW_MUX_CTL register (when available) to "1" and can be used for:

- LoopBack - Module X drives the contact and also receive contact value as an input.
- GPIO Capture - Module X drives the BGA contact and the value is captured by GPIO.

NOTE

Software loopback is not affecting EMI PADS.

A.3.3 Daisy Chain—Multi Pads Ddriving Same Module Input Pin

In some cases, more than one BGA contact may drive a single module input pin.

Such cases require adding one more level of IOMUXing: all these input signals are muxed and a dedicated SW controlled register controls the mux in order to select the required input path. This means that a BGA contact involved in such "Daisy Chain" situation may require 2 SW configuration commands: one for selecting the mode for this contact and one for defining it as the input path.

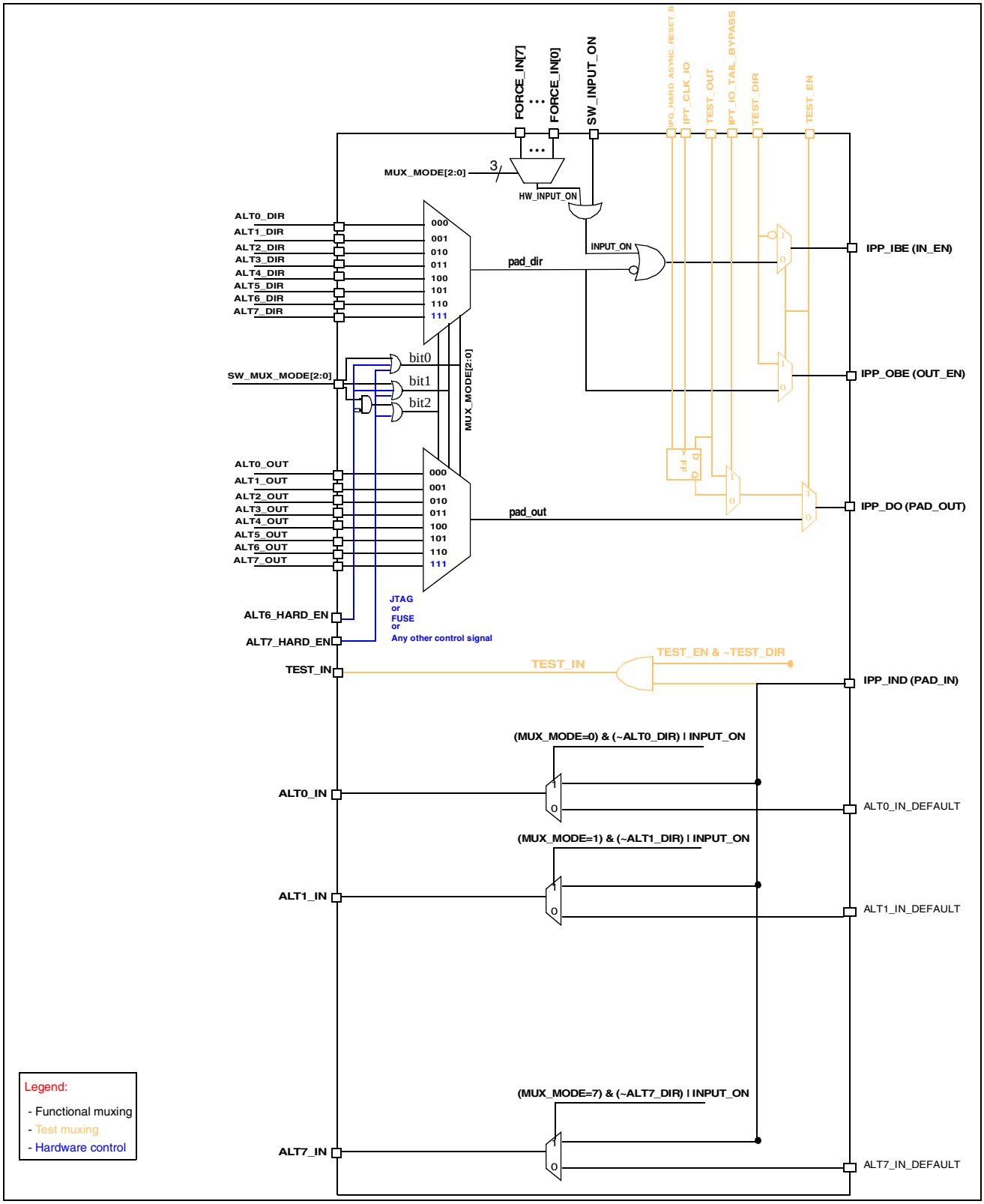


Figure A-3. IOMUX Cell Block Diagram

A.3.4 Interrupts

There are no interrupts that are generated by the IOMUX controller.

A.4 Memory Map and Register Definition

A.4.1 Registers Definition

A.4.1.1 General Purpose Registers

Table A-1. Register: IOMUXC_GPR0

Offset 0x0000 (IOMUXC_GPR0) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	EMI_MUX	0	SDMA_SEL_EV20	SDMA_SEL_EV21	SDMA_SEL_EV23	SDMA_SEL_EV24	SDMA_SEL_EV25	SDMA_SEL_EV35	SDMA_SEL_EV46	SDMA_SEL_EV37	SDMA_SEL_EV47
W							TPIU_TRACE_EN									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-2. Register IOMUXC_GPR0 Bits Description

Field	Description
31-11 GPR	Reserved.
10 EMI_MUX	0 - EMIv2.1 DMA access 1 request. 1 - EMIv2.1 snooping 2 request to IPUv3EX.
9 TPIU_TRACE_EN	Special enable for ALT6 for PADS EIM_D16 - EIM_D31. 0 - PADS acts as defined by their MUX_MODE configuration. 1 - PADS MUX_MODE is switched to ALT6 and connected to TPIU.TRACEDATA[15:0].
8 SDMA_SEL_EV20	0 - Selects ESDHCv2 1 DMA request for SDMA event 20. 1 - Selects I2C1 DMA request for SDMA event 20.
7 SDMA_SEL_EV21	0 - Selects ESDHCv2 2 DMA request for SDMA event 21. 1 - Selects I2C2 DMA request for SDMA event 21.
6 SDMA_SEL_EV23	Reserved.
5 SDMA_SEL_EV24	Reserved.

Table A-2. Register IOMUXC_GPR0 Bits Description (continued)

Field	Description
4 SDMA_SEL_EV25	Reserved.
3 SDMA_SEL_EV35	Reserved.
2 SDMA_SEL_EV46	Reserved.
1 SDMA_SEL_EV37	Reserved.
0 SDMA_SEL_EV47	Reserved.

Table A-3. Register: IOMUXC_GPR1

Offset 0x0004 (IOMUXC_GPR1) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GPR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-4. Register IOMUXC_GPR1 Bits Description

Field	Description
31-0 GPR	General Purpose bits to be used by SoC integration.

Table A-5. Register: IOMUXC_OBSERVE_MUX_0

Offset 0x0008 (IOMUXC_OBSERVE_MUX_0) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	OBSRV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-6. Register IOMUXC_OBSERVE_MUX_0 Bits Description

Field	Description
31-6	Reserved

Table A-6. Register IOMUXC_OBSERVE_MUX_0 Bits Description (continued)

Field	Description
5-0 OBSRV	Select Instance Pin for Observability IOMUXC_OBSERVE_MUX_0 000000: Select Instance ahbmax, Pin max_halted 000001: Select Instance ccm, Pin ccm_clk_switch_ack 000010: Select Instance ccm, Pin ccm_ipg_stop 000011: Select Instance ccm, Pin ccm_ipg_wait 000100: Select Instance ccm, Pin ccm_lpsr_ipu 000101: Select Instance ccm, Pin ccm_pdn_4all_req 000110: Select Instance ccm, Pin hndsk_current_state[0] 000111: Select Instance ccm, Pin ipi_int_1 001000: Select Instance ccm, Pin ipi_int_2 001001: Select Instance ccm, Pin lpm_current_state[0] 001010: Select Instance ccm, Pin shd_current_state[0] 001011: Select Instance cspi, Pin ~ipi_int_cspi_b 001100: Select Instance csu, Pin ~ipi_int_csu_b 001101: Select Instance dpllip1, Pin dpllip_cpen 001110: Select Instance ecspi1, Pin ipd_req_cspi_rdma_b 001111: Select Instance ecspi1, Pin ipd_req_cspi_tdma_b 010000: Select Instance ecspi1, Pin ~ipi_int_cspi_b 010001: Select Instance ecspi2, Pin ipd_req_cspi_rdma_b 010010: Select Instance ecspi2, Pin ~ipi_int_cspi_b 010011: Select Instance emi, Pin chosen_data_master_fast[0] 010100: Select Instance emi, Pin chosen_data_master_int2[0] 010101: Select Instance emi, Pin chosen_data_master_int1[0] 010110: Select Instance emi, Pin chosen_data_master_slow[0] 010111: Select Instance emi, Pin dvfs_ack_fast 011000: Select Instance emi, Pin dvfs_ack_int1 011001: Select Instance emi, Pin ~ipi_emi_ap_int_b 011010: Select Instance emi, Pin ~ipi_int_nfc_b 011011: Select Instance emi, Pin ipp_obe_data_dir[3] 011100: Select Instance emi, Pin ipp_obe_maddr_dir[1] 011101: Select Instance emi, Pin lpack 011110: Select Instance emi, Pin lpmd_int1_s1_mem 011111: Select Instance epit1, Pin ipi_int_epit_oc

Table A-6. Register IOMUXC_OBSERVE_MUX_0 Bits Description (continued)

Field	Description
5-0 OBSRV	100000: Select Instance epit2, Pin ipi_int_epit_oc 100001: Select Instance esdhc1, Pin ~ipi_esdhcv2_irq_b 100010: Select Instance esdhc2, Pin ~ipi_esdhcv2_irq_b 100011: Select Instance esdhc3, Pin ~ipi_esdhcv2_irq_b 100100: Select Instance esdhc4, Pin ~ipi_esdhcv2_irq_b 100101: Select Instance fec, Pin fec_ipi_int 100110: Select Instance firi, Pin ~ipi_int_b 100111: Select Instance gpc, Pin gpc_cta8_pg[0] 101000: Select Instance gpc, Pin gpc_cta8_pg[1] 101001: Select Instance gpc, Pin gpc_cta8_pg[2] 101010: Select Instance gpc, Pin gpc_cta8_pg[3] 101011: Select Instance gpc, Pin gpc_cta8_pg[4] 101100: Select Instance gpc, Pin gpc_emi_pg[0] 101101: Select Instance gpc, Pin gpc_emi_short_b 101110: Select Instance gpc, Pin gpc_event 101111: Select Instance gpc, Pin gpc_int 110000: Select Instance gpc, Pin gpc_int2 110001: Select Instance gpc, Pin gpc_neon_pg[0] 110010: Select Instance pata, Pin ata_rcv_fifo_alarm 110011: Select Instance pata, Pin ata_tx_fifo_alarm 110100: Select Instance pata, Pin ata_txfer_end_alarm 110101: Select Instance slm, Pin ipd_tx_smc_req 110110: Select Instance src, Pin arm_por_rst 110111: Select Instance src, Pin warm_reset 111000: Select Instance tigerp_platform_ne_32k_256k, Pin dsm_request 111001: Select Instance tigerp_platform_ne_32k_256k, Pin ~nm_irq_b 111010: Select Instance tzic, Pin tzic_fiq_b 111011: Select Instance tzic, Pin tzic_irq_b 111100: Select Instance vpu, Pin ipi_int_vpu

Table A-7. Register: IOMUXC_OBSERVE_MUX_1

Offset 0x000c (IOMUXC_OBSERVE_MUX_1) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	OBSRV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-8. Register IOMUXC_OBSERVE_MUX_1 Bits Description

Field	Description
31-6	Reserved

Table A-8. Register IOMUXC_OBSERVE_MUX_1 Bits Description (continued)

Field	Description
5-0 OBSRV	Select Instance Pin for Observability IOMUXC_OBSERVE_MUX_1 000000: Select Instance ccm, Pin ahbmax_halt_req 000001: Select Instance ccm, Pin ccm_pdn_4arm_req 000010: Select Instance ccm, Pin ccm_pup_req 000011: Select Instance ccm, Pin ccm_system_in_stop_mode 000100: Select Instance ccm, Pin ccm_system_in_wait_mode 000101: Select Instance ccm, Pin dpll_en_dpllip 000110: Select Instance ccm, Pin emi_dvfs_req_fast 000111: Select Instance ccm, Pin emi_dvfs_req_int1 001000: Select Instance ccm, Pin emi_dvfs_req_slow 001001: Select Instance ccm, Pin emi_lpmd 001010: Select Instance ccm, Pin emi_lpmd_fast 001011: Select Instance ccm, Pin hndsk_current_state[1] 001100: Select Instance ccm, Pin lpm_current_state[1] 001101: Select Instance ccm, Pin shd_current_state[1] 001110: Select Instance dpllip2, Pin dpllip_cpen 001111: Select Instance ecspi2, Pin ipd_req_cspi_tdma_b 010000: Select Instance emi, Pin chosen_data_master_fast[1] 010001: Select Instance emi, Pin chosen_data_master_int2[1] 010010: Select Instance emi, Pin chosen_data_master_int1[1] 010011: Select Instance emi, Pin chosen_data_master_slow[1] 010100: Select Instance emi, Pin dvfs_ack 010101: Select Instance emi, Pin ipp_obe_data_dir[2] 010110: Select Instance emi, Pin ipp_obe_maddr_dir[0] 010111: Select Instance emi, Pin lpack_fast 011000: Select Instance emi, Pin lpmd_int1_s0_mem 011001: Select Instance firi, Pin dma_req_b[0] 011010: Select Instance firi, Pin dma_req_b[1] 011011: Select Instance gpc, Pin gpc_cta8_clk_upd_req 011100: Select Instance gpc, Pin gpc_cta8_pg[5] 011101: Select Instance gpc, Pin gpc_cta8_pg[6] 011110: Select Instance gpc, Pin gpc_cta8_pg[7] 011111: Select Instance gpc, Pin gpc_cta8_pg[8] ct Instance wdog1, Pin wdog_rst_b

Table A-8. Register IOMUXC_OBSERVE_MUX_1 Bits Description (continued)

Field	Description
5-0 OBSRV	100000: Select Instance gpc, Pin gpc_cta8_pg[9] 100001: Select Instance gpc, Pin gpc_emi_pg[1] 100010: Select Instance gpc, Pin gpc_ipu_switch_b 100011: Select Instance gpc, Pin gpc_l1bits_pwrdwn 100100: Select Instance gpc, Pin gpc_l2bits_pwrdwn 100101: Select Instance gpc, Pin gpc_neon_pg[1] 100110: Select Instance gpio1, Pin ipi_gpio_int15_0 100111: Select Instance gpio1, Pin ipi_gpio_int31_16 101000: Select Instance gpio1, Pin ipi_gpio_int32[0] 101001: Select Instance gpio1, Pin ipi_gpio_int32[1] 101010: Select Instance gpio1, Pin ipi_gpio_int32[2] 101011: Select Instance gpio1, Pin ipi_gpio_int32[3] 101100: Select Instance gpio1, Pin ipi_gpio_int32[4] 101101: Select Instance gpio1, Pin ipi_gpio_int32[5] 101110: Select Instance gpio1, Pin ipi_gpio_int32[6] 101111: Select Instance gpio1, Pin ipi_gpio_int32[7] 110000: Select Instance gpio2, Pin ipi_gpio_int15_0 110001: Select Instance gpio2, Pin ipi_gpio_int31_16 110010: Select Instance gpio3, Pin ipi_gpio_int15_0 110011: Select Instance gpio3, Pin ipi_gpio_int31_16 110100: Select Instance gpio4, Pin ipi_gpio_int15_0 110101: Select Instance gpio4, Pin ipi_gpio_int31_16 110110: Select Instance gpt, Pin ipi_int_gpt 110111: Select Instance gpu3d, Pin ~gpu_int_b 111000: Select Instance src, Pin arm_soc_rst_b 111001: Select Instance tigerp_platform_ne_32k_256k, Pin dsm_request 111010: Select Instance uart1, Pin ipd_uart_rx_dmareq_b 111011: Select Instance uart1, Pin ipd_uart_tx_dmareq_b 111100: Select Instance uart2, Pin ipd_uart_rx_dmareq_b 111101: Select Instance uart2, Pin ipd_uart_tx_dmareq_b 111110: Sele

Table A-9. Register: IOMUXC_OBSERVE_MUX_2

Offset 0x0010 (IOMUXC_OBSERVE_MUX_2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	OBSRV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-10. Register IOMUXC_OBSERVE_MUX_2 Bits Description

Field	Description
31-6	Reserved
5-0 OBSRV	Select Instance Pin for Observability IOMUXC_OBSERVE_MUX_2 000000: Select Instance ccm, Pin emi_freq_change_req 000001: Select Instance ccm, Pin emi_lpmd_int1 000010: Select Instance ccm, Pin hndsk_current_state[2] 000011: Select Instance ccm, Pin ipu_clk_changed 000100: Select Instance ccm, Pin lpm_current_state[2] 000101: Select Instance dpllip3, Pin dpllip_cpen 000110: Select Instance emi, Pin chosen_data_master_fast[2] 000111: Select Instance emi, Pin chosen_data_master_int2[2] 001000: Select Instance emi, Pin chosen_data_master_int1[2] 001001: Select Instance emi, Pin chosen_data_master_slow[2] 001010: Select Instance emi, Pin ipp_obe_weim_nfc_dir[0] 001011: Select Instance emi, Pin lpack 001100: Select Instance emi, Pin lpack_fast 001101: Select Instance emi, Pin lpack_int1 001110: Select Instance emi, Pin lpack_slow 001111: Select Instance emi, Pin lpmd_int1_s1_mem 010000: Select Instance gpc, Pin gpc_core_pwrdwn 010001: Select Instance gpc, Pin gpc_cta8_clk_dvfs_operation 010010: Select Instance gpc, Pin gpc_cta8_pg[10] 010011: Select Instance gpc, Pin gpc_cta8_pg[11] 010100: Select Instance gpc, Pin gpc_cta8_pg[12] 010101: Select Instance gpc, Pin gpc_cta8_pg[13] 010110: Select Instance gpc, Pin gpc_cta8_pg[14] 010111: Select Instance gpc, Pin gpc_emi_pg[2] 011000: Select Instance gpc, Pin gpc_neon_pg[2] 011001: Select Instance gpc, Pin gpc_neon_pwrdwn 011010: Select Instance gpc, Pin gpc_pdn_ack 011011: Select Instance gpc, Pin gpc_pup_ack 011100: Select Instance gpu3d, Pin ~gpu_idle 011101: Select Instance hsc_mipi_mix, Pin mcg_err_int 011110: Select Instance hsc_mipi_mix, Pin mcg_func_int 011111: Select Instance hsc_mipi_mix, Pin mcg_tmr_int 100000: Select Instance i2c1, Pin ~ipi_int_b 100001: Select Instance i2c2, Pin ~ipi_int_b 100010: Select Instance iim, Pin ~iim_irq_b 100011: Select Instance ipu, Pin ipi_int_ipu_err 100100: Select Instance ipu, Pin ipi_int_ipu_func 100101: Select Instance kpp, Pin ~ipi_int_kpp_b 100110: Select Instance owire, Pin ipi_owire_int 100111: Select Instance pata, Pin ipbus_int 101000: Select Instance pwm1, Pin ipi_int_pwm 101001: Select Instance pwm2, Pin ipi_int_pwm 101010: Select Instance rtic, Pin ipi_rtic_done_int 101011: Select Instance sahara, Pin ~ipi_int_sahara_host0_b 101100: Select Instance src, Pin sjc_por_rst_b 101101: Select Instance src, Pin system_rst_b 101110: Select Instance ssi1, Pin ipd_ssi_rx0_dmareq_b 101111: Select Instance ssi1, Pin ipd_ssi_rx1_dmareq_b 110000: Select Instance ssi1, Pin ipd_ssi_tx1_dmareq_b 110001: Select Instance ssi2, Pin ipd_ssi_rx1_dmareq_b

Table A-11. Register: IOMUXC_OBSERVE_MUX_3

Offset	0x0014 (IOMUXC_OBSERVE_MUX_3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	OBSRV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-12. Register IOMUXC_OBSERVE_MUX_3 Bits Description

Field	Description
31-6	Reserved
5-0 OBSRV	Select Instance Pin for Observability IOMUXC_OBSERVE_MUX_3 000000: Select Instance ccm, Pin emi_lpmd_slow 000001: Select Instance ccm, Pin rtic_ipg_stop_req 000010: Select Instance cspi, Pin ipd_req_cspi_rdma_b 000011: Select Instance cspi, Pin ipd_req_cspi_tdma_b 000100: Select Instance dpllip1, Pin dpllip_lrf_sticky 000101: Select Instance dpllip2, Pin dpllip_lrf_sticky 000110: Select Instance dpllip3, Pin dpllip_lrf_sticky 000111: Select Instance emi, Pin dvfs_ack_slow 001000: Select Instance emi, Pin ipp_obe_weim_nfc_dir[1] 001001: Select Instance emi, Pin lpack_slow 001010: Select Instance emi, Pin nfc_dma_rd_req 001011: Select Instance emi, Pin nfc_dma_wr_req 001100: Select Instance emi, Pin weim_idle 001101: Select Instance epit2, Pin ipi_int_epit_oc 001110: Select Instance gpc, Pin gpc_cta8_pg[15] 001111: Select Instance gpc, Pin gpc_cta8_pg[16] 010000: Select Instance gpc, Pin gpc_cta8_pg[17] 010001: Select Instance gpc, Pin gpc_cta8_pg[18] 010010: Select Instance gpc, Pin gpc_cta8_pg[19] 010011: Select Instance gpc, Pin gpc_emi_pg[3] 010100: Select Instance gpc, Pin gpc_freq_change_mode 010101: Select Instance gpc, Pin gpc_neon_pg[3] 010110: Select Instance gpc, Pin gpc_neon_short_b 010111: Select Instance gpu3d, Pin gpu_idle 011000: Select Instance ipu, Pin ipg_clk_change_ack 011001: Select Instance ipu, Pin ipu_sdma_event 011010: Select Instance ipu, Pin ipu_stby_ack 011011: Select Instance sahara, Pin ipg_stop_ack 011100: Select Instance scc, Pin ipg_stop_ack 011101: Select Instance scc, Pin scc_ipi_sctl_irq 011110: Select Instance scc, Pin scc_ipi_sctl_ns_irq 011111: Select Instance scc, Pin scc_ipi_smon_irq 100000: Select Instance sdma, Pin ipg_stop_ack 100001: Select Instance sdma, Pin ~ipi_host_intr_b 100010: Select Instance sim, Pin ipi_int_sim_data_irq 100011: Select Instance sim, Pin ipi_int_sim_ipb_int 100100: Select Instance slm, Pin ~ipi_slm_excep_int_b 100101: Select Instance slm, Pin ~ipi_slm_int_b 100110: Select Instance slm, Pin ~ipi_slm_rx_smc_int_b 100111: Select Instance spdif, Pin ~mirq_tx_b 101000: Select Instance src, Pin any_pu_rst_b 101001: Select Instance src, Pin emi_rst_b 101010: Select Instance src, Pin ipi_int_1 101011: Select Instance src, Pin system_early_rst_b 101100: Select Instance srtc, Pin ~ipi_srtc_int_b 101101: Select Instance srtc, Pin ~ipi_srtc_sec_int_b 101110: Select Instance ssi1, Pin ipd_ssi_tx0_dmareq_b 101111: Select Instance ssi1, Pin ~ipi_int_b 110000: Select Instance ssi2, Pin ~ipi_int_b 110001: Select Instance ssi3, Pin ~ipi_int_b 110010: Select Instance tigerp_platform_ne_32k_256k, Pin nm_irq_b

Table A-13. Register: IOMUXC_OBSERVE_MUX_4

Offset: 0x0018 (IOMUXC_OBSERVE_MUX_4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	OBSRV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-14. Register IOMUXC_OBSERVE_MUX_4 Bits Description

Field	Description
31-6	Reserved

Table A-14. Register IOMUXC_OBSERVE_MUX_4 Bits Description

Field	Description
5-0 OBSRV	Select Instance Pin for Observability IOMUXC_OBSERVE_MUX_4 000000: Select Instance ccm, Pin emi_lpmd_garb 000001: Select Instance ccm, Pin ipu_freq_change_req 000010: Select Instance ccm, Pin ipu_stop_clk_at_stop_req 000011: Select Instance ccm, Pin pll_lvs 000100: Select Instance ccm, Pin sahara_ipg_stop_req 000101: Select Instance ccm, Pin scc_ipg_stop_req 000110: Select Instance ccm, Pin sdma_ipg_stop_req 000111: Select Instance ccm, Pin src_clock_ready 001000: Select Instance emi, Pin esdctl_idle 001001: Select Instance emi, Pin nfc_idle 001010: Select Instance esdhc3, Pin ipd_esdhcv2_dreq_b 001011: Select Instance esdhc4, Pin ipd_esdhcv2_dreq_b 001100: Select Instance gpc, Pin gpc_cta8_iso 001101: Select Instance gpc, Pin gpc_cta8_pg[20] 001110: Select Instance gpc, Pin gpc_emi_pg[4] 001111: Select Instance gpc, Pin gpc_ipu_pg_event 010000: Select Instance gpc, Pin gpc_ipu_stat_pg 010001: Select Instance gpc, Pin volt_chng 010010: Select Instance iim, Pin fuse_latched 010011: Select Instance ipu, Pin ipu_stby_ack 010100: Select Instance rtic, Pin ipg_stop_ack 010101: Select Instance sfp, Pin sfp_ready 010110: Select Instance slm, Pin ipd_tx_req_4 010111: Select Instance spdif, Pin drq1_spdif_b 011000: Select Instance src, Pin emi_dvfs_req 011001: Select Instance src, Pin en_system_clk 011010: Select Instance src, Pin memory_repair_mode 011011: Select Instance src, Pin power_gating_reset_done 011100: Select Instance tigerp_platform_ne_32k_256k, Pin cti1_trigout0 011101: Select Instance tigerp_platform_ne_32k_256k, Pin cti1_trigout1 011110: Select Instance tigerp_platform_ne_32k_256k, Pin cti1_trigout2 011111: Select Instance tigerp_platform_ne_32k_256k, Pin cti1_trigout3 100000: Select Instance tigerp_platform_ne_32k_256k, Pin ~cti_irq_b 100001: Select Instance tigerp_platform_ne_32k_256k, Pin dbgack 100010: Select Instance tigerp_platform_ne_32k_256k, Pin ~pmu_irq_b 100011: Select Instance tve, Pin ipi_tve_int 100100: Select Instance tzic, Pin tzic_wakeup_request 100101: Select Instance uart1, Pin ~ipi_uart_anded_b 100110: Select Instance uart2, Pin ~ipi_uart_anded_b 100111: Select Instance uart3, Pin ipd_uart_rx_dmareq_b 101000: Select Instance uart3, Pin ipd_uart_tx_dmareq_b 101001: Select Instance uart3, Pin ~ipi_uart_anded_b 101010: Select Instance usboh3, Pin ipi_int_uh1 101011: Select Instance usboh3, Pin ipi_int_uh2 101100: Select Instance usboh3, Pin ipi_int_uh3 101101: Select Instance usboh3, Pin ipi_int_uotg 101110: Select Instance vpu, Pin ipi_int_vpu 101111: Select Instance vpu, Pin vpu_idle 110000: Select Instance wdog1, Pin ~ipi_wdog_int_b 110001: Select Instance wdog2, Pin ~ipi_wdog_int_b 110010: Select Instance wdog2, Pin wdog_rst_b 110011: Select Instance hsi2c, Pin ~ipi_hsi2c_int_b 110100: Select Instance sahara, Pin ~ipi_int_sahara_host1_b

Table A-15. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA0

Offset: 0x001c (IOMUXC_SW_MUX_CTL_PAD_EIM_DA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-16. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA0 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA0. 00: Select mux mode: ALT0 mux port: EIM_DA[0] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[16] of instance: tpiu.

Table A-17. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA1

Offset: 0x0020 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-18. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA1 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA1. 00: Select mux mode: ALT0 mux port: EIM_DA[1] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[17] of instance: tpiu.

Table A-19. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA2

Offset 0x0024 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA2) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-20. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA2 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA2. 00: Select mux mode: ALT0 mux port: EIM_DA[2] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[18] of instance: tpiu.

Table A-21. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA3

Offset: 0x0028 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-22. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA3 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA3. 00: Select mux mode: ALT0 mux port: EIM_DA[3] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[19] of instance: tpiu.

Table A-23. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA4

Offset: 0x002c (IOMUXC_SW_MUX_CTL_PAD_EIM_DA4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-24. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA4 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA4. 00: Select mux mode: ALT0 mux port: EIM_DA[4] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[20] of instance: tpiu.

Table A-25. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA5

Offset: 0x0030 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-26. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA5 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA5. 00: Select mux mode: ALT0 mux port: EIM_DA[5] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[21] of instance: tpiu.

Table A-27. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA6

Offset	0x0034 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-28. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA6 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA6. 00: Select mux mode: ALT0 mux port: EIM_DA[6] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[22] of instance: tpiu.

Table A-29. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA7

Offset: 0x0038 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-30. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA7 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA7. 00: Select mux mode: ALT0 mux port: EIM_DA[7] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[23] of instance: tpiu.

Table A-31. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA8

Offset: 0x003c (IOMUXC_SW_MUX_CTL_PAD_EIM_DA8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-32. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA8 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA8. 00: Select mux mode: ALT0 mux port: EIM_DA[8] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[24] of instance: tpiu.

Table A-33. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA9

Offset: 0x0040 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-34. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA9 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA9. 00: Select mux mode: ALT0 mux port: EIM_DA[9] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[25] of instance: tpiu.

Table A-35. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA10

Offset: 0x0044 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-36. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA10 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA10. 00: Select mux mode: ALT0 mux port: EIM_DA[10] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[26] of instance: tpiu.

Table A-37. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA11

Offset: 0x0048 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-38. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA11 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA11. 00: Select mux mode: ALT0 mux port: EIM_DA[11] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[27] of instance: tpiu.

Table A-39. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA12

Offset 0x004c (IOMUXC_SW_MUX_CTL_PAD_EIM_DA12) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-40. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA12 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA12. 00: Select mux mode: ALT0 mux port: EIM_DA[12] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[28] of instance: tpiu.

Table A-41. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA13

Offset	0x0050 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-42. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA13 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA13. 00: Select mux mode: ALT0 mux port: EIM_DA[13] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[29] of instance: tpiu.

Table A-43. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA14

Offset	0x0054 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-44. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA14 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA14. 00: Select mux mode: ALT0 mux port: EIM_DA[14] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[30] of instance: tpiu.

Table A-45. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DA15

Offset	0x0058 (IOMUXC_SW_MUX_CTL_PAD_EIM_DA15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-46. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DA15 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DA15. 00: Select mux mode: ALT0 mux port: EIM_DA[15] of instance: emi. 01: Select mux mode: ALT1 mux port: TRACE[31] of instance: tpiu.

Table A-47. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D16

Offset: 0x005c (IOMUXC_SW_MUX_CTL_PAD_EIM_D16) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-48. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D16 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D16. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D16. 000: Select mux mode: ALT0 mux port: WEIM_D[16] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[0] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA0 of instance: usboh3. 011: Select mux mode: ALT3 mux port: CTS of instance: uart2. 100: Select mux mode: ALT4 mux port: SDA of instance: i2c1. 101: Select mux mode: ALT5 mux port: AUD4_RXFS of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[0] of instance: tpiu. 111: Select mux mode: ALT7 mux port: AUD5_TXD of instance: audmux. NOTE: BGA contact EIM_D16 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT3.

Table A-49. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D17

Offset	0x0060 (IOMUXC_SW_MUX_CTL_PAD_EIM_D17)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-50. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D17 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D17. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D17. 000: Select mux mode: ALT0 mux port: WEIM_D[17] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[1] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA1 of instance: usboh3. 011: Select mux mode: ALT3 mux port: RXD_MUX of instance: uart2. 100: Select mux mode: ALT4 mux port: CTS of instance: uart3. 101: Select mux mode: ALT5 mux port: SISG[4] of instance: ipu. 110: Select mux mode: ALT6 mux port: TRACE[1] of instance: tpiu. 111: Select mux mode: ALT7 mux port: AUD5_RXD of instance: audmux. NOTE: BGA contact EIM_D17 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT3. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT4.

Table A-51. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D18

Offset: 0x0064 (IOMUXC_SW_MUX_CTL_PAD_EIM_D18) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-52. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D18 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D18. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D18. 000: Select mux mode: ALT0 mux port: WEIM_D[18] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[2] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA2 of instance: usboh3. 011: Select mux mode: ALT3 mux port: TXD_MUX of instance: uart2. 100: Select mux mode: ALT4 mux port: RTS of instance: uart3. 101: Select mux mode: ALT5 mux port: SISG[5] of instance: ipu. 110: Select mux mode: ALT6 mux port: TRACE[2] of instance: tpiu. 111: Select mux mode: ALT7 mux port: AUD5_TXC of instance: audmux. NOTE: BGA contact EIM_D18 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT3. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT4.

Table A-53. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D19

Offset	0x0068 (IOMUXC_SW_MUX_CTL_PAD_EIM_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-54. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D19 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D19. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D19. 000: Select mux mode: ALT0 mux port: WEIM_D[19] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[3] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA3 of instance: usboh3. 011: Select mux mode: ALT3 mux port: RTS of instance: uart2. 100: Select mux mode: ALT4 mux port: SCL of instance: i2c1. 101: Select mux mode: ALT5 mux port: AUD4_RXC of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[3] of instance: tpiu. 111: Select mux mode: ALT7 mux port: AUD5_TXFS of instance: audmux. NOTE: BGA contact EIM_D19 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT3.

Table A-55. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D20

Offset	0x006c (IOMUXC_SW_MUX_CTL_PAD_EIM_D20)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-56. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D20 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D20. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D20. 000: Select mux mode: ALT0 mux port: WEIM_D[20] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[4] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA4 of instance: usboh3. 011: Select mux mode: ALT3 mux port: CSU_INT_DEB of instance: csu. 100: Select mux mode: ALT4 mux port: SRTC_ALARM_DEB of instance: srtc. 101: Select mux mode: ALT5 mux port: AUD4_TXD of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[4] of instance: tpiu. 111: Select mux mode: ALT7 mux port: TXREADY of instance: usbphy. NOTE: BGA contact EIM_D20 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_DB_AMX_SELECT_INPUT for mode ALT5.

Table A-57. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D21

Offset	0x0070 (IOMUXC_SW_MUX_CTL_PAD_EIM_D21)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-58. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D21 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D21. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_D21. 000: Select mux mode: ALT0 mux port: WEIM_D[21] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[5] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA5 of instance: usboh3. 011: Select mux mode: ALT3 mux port: SRTC_ALARM_DEB of instance: srtc. 101: Select mux mode: ALT5 mux port: AUD4_RXD of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[5] of instance: tpiu. 111: Select mux mode: ALT7 mux port: RXVALID of instance: usbphy. NOTE: BGA contact EIM_D21 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_DA_AMX_SELECT_INPUT for mode ALT5.

Table A-59. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D22

Offset	0x0074 (IOMUXC_SW_MUX_CTL_PAD_EIM_D22)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-60. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D22 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D22. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_D22. 000: Select mux mode: ALT0 mux port: WEIM_D[22] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[6] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA6 of instance: usboh3. 011: Select mux mode: ALT3 mux port: FAIL_STATE of instance: scc. 101: Select mux mode: ALT5 mux port: AUD4_TXC of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[6] of instance: tpiu. 111: Select mux mode: ALT7 mux port: RXACTIVE of instance: usbphy. NOTE: BGA contact EIM_D22 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT5.

Table A-61. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D23

Offset	0x0078 (IOMUXC_SW_MUX_CTL_PAD_EIM_D23)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-62. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D23 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D23. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D23. 000: Select mux mode: ALT0 mux port: WEIM_D[23] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[7] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DATA7 of instance: usboh3. 011: Select mux mode: ALT3 mux port: SEC_STATE of instance: scc. 100: Select mux mode: ALT4 mux port: OUT1 of instance: spdif. 101: Select mux mode: ALT5 mux port: AUD4_TXFS of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[7] of instance: tpiu. 111: Select mux mode: ALT7 mux port: RXERROR of instance: usbphy. NOTE: BGA contact EIM_D23 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT5.

Table A-63. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D24

Offset: 0x007c (IOMUXC_SW_MUX_CTL_PAD_EIM_D24) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-64. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D24 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D24. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D24. 000: Select mux mode: ALT0 mux port: WEIM_D[24] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[8] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_DATA0 of instance: usboh3. 011: Select mux mode: ALT3 mux port: CTS of instance: uart3. 100: Select mux mode: ALT4 mux port: SDA of instance: i2c2. 101: Select mux mode: ALT5 mux port: AUD6_RXFS of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[8] of instance: tpiu. 111: Select mux mode: ALT7 mux port: SIECLOCK of instance: usbphy. NOTE: BGA contact EIM_D24 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT3.

Table A-65. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D25

Offset	0x0080 (IOMUXC_SW_MUX_CTL_PAD_EIM_D25)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-66. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D25 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D25. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D25. 000: Select mux mode: ALT0 mux port: WEIM_D[25] of instance: emi. 001: Select mux mode: ALT1 mux port: COL[6] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA1 of instance: usboh3. 011: Select mux mode: ALT3 mux port: RXD_MUX of instance: uart3. 100: Select mux mode: ALT4 mux port: CTS of instance: uart2. 101: Select mux mode: ALT5 mux port: CMPOUT1 of instance: gpt. 110: Select mux mode: ALT6 mux port: TRACE[9] of instance: tpiu. 111: Select mux mode: ALT7 mux port: VBUSVALID of instance: usbphy. NOTE: BGA contact EIM_D25 is involved in Daisy Chain. - Config Register IOMUXC_KPP_IPP_IND_COL_6_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT3.

Table A-67. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D26

Offset: 0x0084 (IOMUXC_SW_MUX_CTL_PAD_EIM_D26) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-68. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D26 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D26. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D26. 000: Select mux mode: ALT0 mux port: WEIM_D[26] of instance: emi. 001: Select mux mode: ALT1 mux port: COL[7] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA2 of instance: usboh3. 011: Select mux mode: ALT3 mux port: TXD_MUX of instance: uart3. 100: Select mux mode: ALT4 mux port: RTS of instance: uart2. 101: Select mux mode: ALT5 mux port: CMPOUT2 of instance: gpt. 110: Select mux mode: ALT6 mux port: TRACE[10] of instance: tpiu. 111: Select mux mode: ALT7 mux port: AVALID of instance: usbphy. NOTE: BGA contact EIM_D26 is involved in Daisy Chain. - Config Register IOMUXC_KPP_IPP_IND_COL_7_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT3.

Table A-69. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D27

Offset	0x0088 (IOMUXC_SW_MUX_CTL_PAD_EIM_D27)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-70. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D27 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_D27. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D27. 000: Select mux mode: ALT0 mux port: WEIM_D[27] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[9] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_DATA3 of instance: usboh3. 011: Select mux mode: ALT3 mux port: RTS of instance: uart3. 100: Select mux mode: ALT4 mux port: SCL of instance: i2c2. 101: Select mux mode: ALT5 mux port: AUD6_RXC of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[11] of instance: tpiu. 111: Select mux mode: ALT7 mux port: BVALID of instance: usbphy. NOTE: BGA contact EIM_D27 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT3.

Table A-71. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D28

Offset	0x008c (IOMUXC_SW_MUX_CTL_PAD_EIM_D28)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-72. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D28 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D28. 000: Select mux mode: ALT0 mux port: WEIM_D[28] of instance: emi. 001: Select mux mode: ALT1 mux port: ROW[4] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA4 of instance: usboh3. 011: Select mux mode: ALT3 mux port: OBSRV_INT_OUT0 of instance: elvis_observe_mux. 100: Select mux mode: ALT4 mux port: SISG[0] of instance: ipu. 101: Select mux mode: ALT5 mux port: AUD6_TXD of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[12] of instance: tpiu. 111: Select mux mode: ALT7 mux port: ENDSESSION of instance: usbphy. NOTE: BGA contact EIM_D28 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_DB_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_KPP_IPP_IND_ROW_4_SELECT_INPUT for mode ALT1.

Table A-73. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D29

Offset	0x0090 (IOMUXC_SW_MUX_CTL_PAD_EIM_D29)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-74. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D29 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D29. 000: Select mux mode: ALT0 mux port: WEIM_D[29] of instance: emi. 001: Select mux mode: ALT1 mux port: ROW[5] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA5 of instance: usboh3. 011: Select mux mode: ALT3 mux port: OBSRV_INT_OUT1 of instance: elvis_observe_mux. 100: Select mux mode: ALT4 mux port: SISG[1] of instance: ipu. 101: Select mux mode: ALT5 mux port: AUD6_RXD of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[13] of instance: tpiu. 111: Select mux mode: ALT7 mux port: IDDIG of instance: usbphy. NOTE: BGA contact EIM_D29 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_DA_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_KPP_IPP_IND_ROW_5_SELECT_INPUT for mode ALT1.

Table A-75. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D30

Offset	0x0094 (IOMUXC_SW_MUX_CTL_PAD_EIM_D30)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-76. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D30 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D30. 000: Select mux mode: ALT0 mux port: WEIM_D[30] of instance: emi. 001: Select mux mode: ALT1 mux port: ROW[6] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA6 of instance: usboh3. 011: Select mux mode: ALT3 mux port: OBSRV_INT_OUT2 of instance: elvis_observe_mux. 100: Select mux mode: ALT4 mux port: SISG[2] of instance: ipu. 101: Select mux mode: ALT5 mux port: AUD6_TXC of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[14] of instance: tpiu. 111: Select mux mode: ALT7 mux port: HOSTDISCONNECT of instance: usbphy. NOTE: BGA contact EIM_D30 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_KPP_IPP_IND_ROW_6_SELECT_INPUT for mode ALT1.

Table A-77. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_D31

Offset	0x0098 (IOMUXC_SW_MUX_CTL_PAD_EIM_D31)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-78. Register IOMUXC_SW_MUX_CTL_PAD_EIM_D31 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_D31. 000: Select mux mode: ALT0 mux port: WEIM_D[31] of instance: emi. 001: Select mux mode: ALT1 mux port: ROW[7] of instance: kpp. 010: Select mux mode: ALT2 mux port: USBOTG_DATA7 of instance: usboh3. 011: Select mux mode: ALT3 mux port: OBSRV_INT_OUT3 of instance: elvis_observe_mux. 100: Select mux mode: ALT4 mux port: SISG[3] of instance: ipu. 101: Select mux mode: ALT5 mux port: AUD6_TXFS of instance: audmux. 110: Select mux mode: ALT6 mux port: TRACE[15] of instance: tpiu. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact EIM_D31 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_KPP_IPP_IND_ROW_7_SELECT_INPUT for mode ALT1.

Table A-79. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A16

Offset: 0x009c (IOMUXC_SW_MUX_CTL_PAD_EIM_A16) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-80. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A16 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A16. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A16. 000: Select mux mode: ALT0 mux port: EIM_A[16] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[10] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[0] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: OSC_FREQ_SEL[0] of instance: src.

Table A-81. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A17

Offset	0x00a0 (IOMUXC_SW_MUX_CTL_PAD_EIM_A17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-82. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A17 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A17. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A17. 000: Select mux mode: ALT0 mux port: EIM_A[17] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[11] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[1] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: OSC_FREQ_SEL[1] of instance: src.

Table A-83. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A18

Offset: 0x00a4 (IOMUXC_SW_MUX_CTL_PAD_EIM_A18) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-84. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A18 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A18. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A18. 000: Select mux mode: ALT0 mux port: EIM_A[18] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[12] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[2] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: BT_LPB[0] of instance: src.

Table A-85. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A19

Offset	0x00a8 (IOMUXC_SW_MUX_CTL_PAD_EIM_A19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-86. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A19 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A19. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A19. 000: Select mux mode: ALT0 mux port: EIM_A[19] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[13] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[3] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: BT_LPB[1] of instance: src.

Table A-87. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A20

Offset	0x00ac (IOMUXC_SW_MUX_CTL_PAD_EIM_A20)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-88. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A20 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A20. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A20. 000: Select mux mode: ALT0 mux port: EIM_A[20] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[14] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[4] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: BT_UART_SRC[0] of instance: src.

Table A-89. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A21

Offset: 0x00b0 (IOMUXC_SW_MUX_CTL_PAD_EIM_A21) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-90. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A21 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A21. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A21. 000: Select mux mode: ALT0 mux port: EIM_A[21] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[15] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[5] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: BT_UART_SRC[1] of instance: src.

Table A-91. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A22

Offset: 0x00b4 (IOMUXC_SW_MUX_CTL_PAD_EIM_A22) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-92. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A22 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A22. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: EIM_A22. 00: Select mux mode: ALT0 mux port: EIM_A[22] of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[16] of instance: gpio2. 10: Select mux mode: ALT2 mux port: DATA_HS_OUT[6] of instance: hsc_mipi_mix.

Table A-93. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A23

Offset	0x00b8 (IOMUXC_SW_MUX_CTL_PAD_EIM_A23)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-94. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A23 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A23. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A23. 000: Select mux mode: ALT0 mux port: EIM_A[23] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[17] of instance: gpio2. 010: Select mux mode: ALT2 mux port: DATA_HS_OUT[7] of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: BT_HPN_EN of instance: src.

Table A-95. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A24

Offset: 0x00bc (IOMUXC_SW_MUX_CTL_PAD_EIM_A24) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-96. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A24 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A24. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: EIM_A24. 00: Select mux mode: ALT0 mux port: EIM_A[24] of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[18] of instance: gpio2. 10: Select mux mode: ALT2 mux port: USBH2_CLK of instance: usboh3.

Table A-97. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A25

Offset	0x00c0 (IOMUXC_SW_MUX_CTL_PAD_EIM_A25)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-98. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A25 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A25. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: EIM_A25. 000: Select mux mode: ALT0 mux port: EIM_A[25] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[19] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_DIR of instance: usboh3. 110: Select mux mode: ALT6 mux port: DI1_PIN4 of instance: ipu.

Table A-99. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A26

Offset	0x00c4 (IOMUXC_SW_MUX_CTL_PAD_EIM_A26)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-100. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A26 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A26. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: EIM_A26. 000: Select mux mode: ALT0 mux port: EIM_A[26] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[20] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_STP of instance: usboh3. 100: Select mux mode: ALT4 mux port: SISG[0] of instance: ipu. 101: Select mux mode: ALT5 mux port: CSI1_DATA_EN of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: DI2_EXT_CLK of instance: ccm. NOTE: BGA contact EIM_A26 is involved in Daisy Chain. - Config Register IOMUXC_CCM_IPP_DI1_CLK_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS2_DATA_EN_SELECT_INPUT for mode ALT5.

Table A-101. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_A27

Offset	0x00c8 (IOMUXC_SW_MUX_CTL_PAD_EIM_A27)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-102. Register IOMUXC_SW_MUX_CTL_PAD_EIM_A27 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_A27. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_A27. 000: Select mux mode: ALT0 mux port: EIM_A[27] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[21] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBH2_NXT of instance: usboh3. 011: Select mux mode: ALT3 mux port: OBSRV_INT_OUT4 of instance: elvis_observe_mux. 100: Select mux mode: ALT4 mux port: SISG[1] of instance: ipu. 101: Select mux mode: ALT5 mux port: CSI2_DATA_EN of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: DI1_PIN1 of instance: hsc_mipi_mix. NOTE: BGA contact EIM_A27 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_HSC_MIPI_MIX_PAR0_VSYNC_SELECT_INPUT for mode ALT6.

Table A-103. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_EB0

Offset: 0x00cc (IOMUXC_SW_MUX_CTL_PAD_EIM_EB0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-104. Register IOMUXC_SW_MUX_CTL_PAD_EIM_EB0 Bits Description

Field	Description
31-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_EB0. 00: Select mux mode: ALT0 mux port: EIM_EB[0] of instance: emi. 10: Select mux mode: ALT2 mux port: DETECT_D of instance: hsc_mipi_mix.

Table A-105. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_EB1

Offset: 0x00d0 (IOMUXC_SW_MUX_CTL_PAD_EIM_EB1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-106. Register IOMUXC_SW_MUX_CTL_PAD_EIM_EB1 Bits Description

Field	Description
31-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_EB1. 00: Select mux mode: ALT0 mux port: EIM_EB[1] of instance: emi. 10: Select mux mode: ALT2 mux port: DELAY_D of instance: hsc_mipi_mix.

Table A-107. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_EB2

Offset	0x00d4 (IOMUXC_SW_MUX_CTL_PAD_EIM_EB2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-108. Register IOMUXC_SW_MUX_CTL_PAD_EIM_EB2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_EB2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_EB2. 000: Select mux mode: ALT0 mux port: EIM_EB[2] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[22] of instance: gpio2. 010: Select mux mode: ALT2 mux port: TRCTL of instance: tpiu. 011: Select mux mode: ALT3 mux port: MDIO of instance: fec. 100: Select mux mode: ALT4 mux port: SISG[2] of instance: ipu. 101: Select mux mode: ALT5 mux port: CSI1_D[2] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_RXFS of instance: audmux. 111: Select mux mode: ALT7 mux port: CMPOUT1 of instance: gpt. NOTE: BGA contact EIM_EB2 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_FEC_FEC_MDI_SELECT_INPUT for mode ALT3.

Table A-109. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_EB3

Offset	0x00d8 (IOMUXC_SW_MUX_CTL_PAD_EIM_EB3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-110. Register IOMUXC_SW_MUX_CTL_PAD_EIM_EB3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_EB3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: EIM_EB3. 000: Select mux mode: ALT0 mux port: EIM_EB[3] of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[23] of instance: gpio2. 010: Select mux mode: ALT2 mux port: TRCLK of instance: tpiu. 011: Select mux mode: ALT3 mux port: RDATA[1] of instance: fec. 100: Select mux mode: ALT4 mux port: SISG[3] of instance: ipu. 101: Select mux mode: ALT5 mux port: CSI1_D[3] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_RXC of instance: audmux. 111: Select mux mode: ALT7 mux port: CMPOUT2 of instance: gpt. NOTE: BGA contact EIM_EB3 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_FEC_FEC_RDATA_1_SELECT_INPUT for mode ALT3.

Table A-111. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_OE

Offset: 0x00dc (IOMUXC_SW_MUX_CTL_PAD_EIM_OE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-112. Register IOMUXC_SW_MUX_CTL_PAD_EIM_OE Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_OE. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_OE. 00: Select mux mode: ALT0 mux port: EIM_OE of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[24] of instance: gpio2.

Table A-113. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS0

Offset	0x00e0 (IOMUXC_SW_MUX_CTL_PAD_EIM_CS0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-114. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_CS0. 00: Select mux mode: ALT0 mux port: EIM_CS0 of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[25] of instance: gpio2.

Table A-115. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS1

Offset	0x00e4 (IOMUXC_SW_MUX_CTL_PAD_EIM_CS1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-116. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: EIM_CS1. 000: Select mux mode: ALT0 mux port: EIM_CS1 of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[26] of instance: gpio2. 100: Select mux mode: ALT4 mux port: SISG[4] of instance: ipu.

Table A-117. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS2

Offset	0x00e8 (IOMUXC_SW_MUX_CTL_PAD_EIM_CS2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-118. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_CS2. 000: Select mux mode: ALT0 mux port: EIM_CS2 of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[27] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_STP of instance: usboh3. 011: Select mux mode: ALT3 mux port: RDATA[2] of instance: fec. 100: Select mux mode: ALT4 mux port: SISG[5] of instance: ipu. 101: Select mux mode: ALT5 mux port: CSI1_D[4] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_TXD of instance: audmux. NOTE: BGA contact EIM_CS2 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_FEC_FEC_RDATA_2_SELECT_INPUT for mode ALT3.

Table A-119. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS3

Offset	0x00ec (IOMUXC_SW_MUX_CTL_PAD_EIM_CS3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-120. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_CS3. 000: Select mux mode: ALT0 mux port: EIM_CS3 of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[28] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_NXT of instance: usboh3. 011: Select mux mode: ALT3 mux port: RDATA[3] of instance: fec. 100: Select mux mode: ALT4 mux port: SSI_EXT2_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: CSI1_D[5] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_RXD of instance: audmux. NOTE: BGA contact EIM_CS3 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_FEC_FEC_RDATA_3_SELECT_INPUT for mode ALT3.

Table A-121. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS4

Offset	0x00f0 (IOMUXC_SW_MUX_CTL_PAD_EIM_CS4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-122. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_CS4. 000: Select mux mode: ALT0 mux port: EIM_CS4 of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[29] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_CLK of instance: usboh3. 011: Select mux mode: ALT3 mux port: RX_ER of instance: fec. 100: Select mux mode: ALT4 mux port: SSI_EXT1_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: CSI1_D[6] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_TXC of instance: audmux. NOTE: BGA contact EIM_CS4 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT for mode ALT3.

Table A-123. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CS5

Offset: 0x00f4 (IOMUXC_SW_MUX_CTL_PAD_EIM_CS5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-124. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CS5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CS5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: EIM_CS5. 000: Select mux mode: ALT0 mux port: EIM_CS5 of instance: emi. 001: Select mux mode: ALT1 mux port: GPIO[30] of instance: gpio2. 010: Select mux mode: ALT2 mux port: USBOTG_DIR of instance: usboh3. 011: Select mux mode: ALT3 mux port: CRS of instance: fec. 100: Select mux mode: ALT4 mux port: DI1_EXT_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: CSI1_D[7] of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: AUD5_TXFS of instance: audmux. NOTE: BGA contact EIM_CS5 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_CCM_IPP_DI0_CLK_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_CRS_SELECT_INPUT for mode ALT3.

Table A-125. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_DTACK

Offset: 0x00f8 (IOMUXC_SW_MUX_CTL_PAD_EIM_DTACK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-126. Register IOMUXC_SW_MUX_CTL_PAD_EIM_DTACK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_DTACK. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_DTACK. 00: Select mux mode: ALT0 mux port: WEIM_DTACK_B of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[31] of instance: gpio2.

Table A-127. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_LBA

Offset: 0x00fc (IOMUXC_SW_MUX_CTL_PAD_EIM_LBA) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-128. Register IOMUXC_SW_MUX_CTL_PAD_EIM_LBA Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_LBA. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_LBA. 00: Select mux mode: ALT0 mux port: EIM_LBA of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[1] of instance: gpio3. NOTE: BGA contact EIM_LBA is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_1_SELECT_INPUT for mode ALT1.

Table A-129. Register: IOMUXC_SW_MUX_CTL_PAD_EIM_CRE

Offset: 0x0100 (IOMUXC_SW_MUX_CTL_PAD_EIM_CRE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-130. Register IOMUXC_SW_MUX_CTL_PAD_EIM_CRE Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact EIM_CRE. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: EIM_CRE. 00: Select mux mode: ALT0 mux port: EIM_CRE of instance: emi. 01: Select mux mode: ALT1 mux port: GPIO[2] of instance: gpio3. NOTE: BGA contact EIM_CRE is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_2_SELECT_INPUT for mode ALT1.

Table A-131. Register: IOMUXC_SW_MUX_CTL_PAD_DRAM_CS1

Offset: 0x0104 (IOMUXC_SW_MUX_CTL_PAD_DRAM_CS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-132. Register IOMUXC_SW_MUX_CTL_PAD_DRAM_CS1 Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DRAM_CS1. 00: Select mux mode: ALT0 mux port: DRAM_CS1 of instance: emi. 01: Select mux mode: ALT1 mux port: CLKO of instance: ccm.

Table A-133. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_WE_B

Offset: 0x0108 (IOMUXC_SW_MUX_CTL_PAD_NANDF_WE_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-134. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_WE_B Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_WE_B. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: NANDF_WE_B. 000: Select mux mode: ALT0 mux port: NANDF_WE_B of instance: emi. 001: Select mux mode: ALT1 mux port: DIOW of instance: pata. 010: Select mux mode: ALT2 mux port: DAT0 of instance: esdhc3. 011: Select mux mode: ALT3 mux port: GPIO[3] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[0] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_WE_B is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT0_IN_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_3_SELECT_INPUT for mode ALT3.

Table A-135. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RE_B

Offset: 0x010c (IOMUXC_SW_MUX_CTL_PAD_NANDF_RE_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-136. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RE_B Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RE_B. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: NANDF_RE_B. 000: Select mux mode: ALT0 mux port: NANDF_RE_B of instance: emi. 001: Select mux mode: ALT1 mux port: DIOR of instance: pata. 010: Select mux mode: ALT2 mux port: DAT1 of instance: esdhc3. 011: Select mux mode: ALT3 mux port: GPIO[4] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[1] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_RE_B is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT1_IN_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_4_SELECT_INPUT for mode ALT3.

Table A-137. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_ALE

Offset	0x0110 (IOMUXC_SW_MUX_CTL_PAD_NANDF_ALE)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-138. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_ALE Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_ALE. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: NANDF_ALE. 000: Select mux mode: ALT0 mux port: NANDF_ALE of instance: emi. 001: Select mux mode: ALT1 mux port: BUFFER_EN of instance: pata. 011: Select mux mode: ALT3 mux port: GPIO[5] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[2] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_ALE is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_5_SELECT_INPUT for mode ALT3.

Table A-139. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CLE

Offset: 0x0114 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CLE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-140. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CLE Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CLE. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: NANDF_CLE. 000: Select mux mode: ALT0 mux port: NANDF_CLE of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_RESET_B of instance: pata. 011: Select mux mode: ALT3 mux port: GPIO[6] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[3] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_CLE is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_6_SELECT_INPUT for mode ALT3.

Table A-141. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_WP_B

Offset: 0x0118 (IOMUXC_SW_MUX_CTL_PAD_NANDF_WP_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-142. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_WP_B Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_WP_B. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: NANDF_WP_B. 000: Select mux mode: ALT0 mux port: NANDF_WP_B of instance: emi. 001: Select mux mode: ALT1 mux port: DMACK of instance: pata. 010: Select mux mode: ALT2 mux port: DAT2 of instance: esdhc3. 011: Select mux mode: ALT3 mux port: GPIO[7] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[4] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_WP_B is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT2_IN_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_7_SELECT_INPUT for mode ALT3.

Table A-143. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RB0

Offset: 0x011c (IOMUXC_SW_MUX_CTL_PAD_NANDF_RB0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-144. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RB0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RB0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: NANDF_RB0. 000: Select mux mode: ALT0 mux port: NANDF_RB0 of instance: emi. 001: Select mux mode: ALT1 mux port: DMARQ of instance: pata. 010: Select mux mode: ALT2 mux port: DAT3 of instance: esdhc3. 011: Select mux mode: ALT3 mux port: GPIO[8] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[4] of instance: sdma. 101: Select mux mode: ALT5 mux port: SS1 of instance: ecspi2. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_RB0 is involved in Daisy Chain. - Config Register IOMUXC_ECSPI2_IPP_IND_SS_B_1_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_ESDHC3_IPP_DAT3_IN_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_8_SELECT_INPUT for mode ALT3.

Table A-145. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RB1

Offset	0x0120 (IOMUXC_SW_MUX_CTL_PAD_NANDF_RB1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-146. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RB1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RB1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_RB1. 000: Select mux mode: ALT0 mux port: NANDF_RB1 of instance: emi. 001: Select mux mode: ALT1 mux port: IORDY of instance: pata. 010: Select mux mode: ALT2 mux port: RDY of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[9] of instance: gpio3. 100: Select mux mode: ALT4 mux port: CMPOUT2 of instance: gpt. 101: Select mux mode: ALT5 mux port: CMD of instance: esdhc4. 110: Select mux mode: ALT6 mux port: MOSI of instance: cspi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact NANDF_RB1 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT for mode ALT6.

Table A-147. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RB2

Offset: 0x0124 (IOMUXC_SW_MUX_CTL_PAD_NANDF_RB2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-148. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RB2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RB2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_RB2. 000: Select mux mode: ALT0 mux port: NANDF_RB2 of instance: emi. 001: Select mux mode: ALT1 mux port: COL of instance: fec. 010: Select mux mode: ALT2 mux port: SCLK of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[10] of instance: gpio3. 100: Select mux mode: ALT4 mux port: CMPOUT3 of instance: gpt. 101: Select mux mode: ALT5 mux port: DI2_WAIT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: USBH3_NXT of instance: usboh3. 111: Select mux mode: ALT7 mux port: H3_DP of instance: usboh3. NOTE: BGA contact NANDF_RB2 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_COL_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_HSC_MIPI_MIX_PAR1_DI_WAIT_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_NXT_SELECT_INPUT for mode ALT6.

Table A-149. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RB3

Offset: 0x0128 (IOMUXC_SW_MUX_CTL_PAD_NANDF_RB3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-150. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RB3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RB3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_RB3. 000: Select mux mode: ALT0 mux port: NANDF_RB3 of instance: emi. 001: Select mux mode: ALT1 mux port: RX_CLK of instance: fec. 010: Select mux mode: ALT2 mux port: MISO of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[11] of instance: gpio3. 100: Select mux mode: ALT4 mux port: TOG_EN of instance: dpllip1. 101: Select mux mode: ALT5 mux port: DI1_WAIT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: USBH3_CLK of instance: usboh3. 111: Select mux mode: ALT7 mux port: H3_DM of instance: usboh3. NOTE: BGA contact NANDF_RB3 is involved in Daisy Chain. - Config Register IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_CLK_SELECT_INPUT for mode ALT6.

Table A-151. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO_NAND

Offset: 0x012c (IOMUXC_SW_MUX_CTL_PAD_GPIO_NAND) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-152. Register IOMUXC_SW_MUX_CTL_PAD_GPIO_NAND Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO_NAND. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: GPIO_NAND. 00: Select mux mode: ALT0 mux port: GPIO[12] of instance: gpio3. 01: Select mux mode: ALT1 mux port: INTRQ of instance: pata. NOTE: BGA contact GPIO_NAND is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_12_SELECT_INPUT for mode ALT0.

Table A-153. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS0

Offset: 0x0130 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-154. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: NANDF_CS0. 000: Select mux mode: ALT0 mux port: NANDF_CS0 of instance: emi. 011: Select mux mode: ALT3 mux port: GPIO[16] of instance: gpio3. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved.

Table A-155. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS1

Offset: 0x0134 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-156. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: NANDF_CS1. 000: Select mux mode: ALT0 mux port: NANDF_CS1 of instance: emi. 011: Select mux mode: ALT3 mux port: GPIO[17] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[5] of instance: sdma. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved.

Table A-157. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS2

Offset: 0x0138 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-158. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_CS2. 000: Select mux mode: ALT0 mux port: NANDF_CS2 of instance: emi. 001: Select mux mode: ALT1 mux port: CS_0 of instance: pata. 010: Select mux mode: ALT2 mux port: TX_ER of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[18] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[5] of instance: sdma. 101: Select mux mode: ALT5 mux port: CLK of instance: esdhc4. 110: Select mux mode: ALT6 mux port: SCLK of instance: cspi. 111: Select mux mode: ALT7 mux port: H1_DP of instance: usboh3. NOTE: BGA contact NANDF_CS2 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT for mode ALT6.

Table A-159. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS3

Offset: 0x013c (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-160. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_CS3. 000: Select mux mode: ALT0 mux port: NANDF_CS3 of instance: emi. 001: Select mux mode: ALT1 mux port: CS_1 of instance: pata. 010: Select mux mode: ALT2 mux port: MDC of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[19] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[6] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT0 of instance: esdhc4. 110: Select mux mode: ALT6 mux port: PD0 of instance: sim. 111: Select mux mode: ALT7 mux port: H1_DM of instance: usboh3.

Table A-161. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS4

Offset	0x0140 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-162. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_CS4. 000: Select mux mode: ALT0 mux port: NANDF_CS4 of instance: emi. 001: Select mux mode: ALT1 mux port: DA_0 of instance: pata. 010: Select mux mode: ALT2 mux port: TDATA[1] of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[20] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVT_CHN_LINES[7] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT1 of instance: esdhc4. 110: Select mux mode: ALT6 mux port: CLK0 of instance: sim. 111: Select mux mode: ALT7 mux port: USBH3_STP of instance: usboh3. NOTE: BGA contact NANDF_CS4 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_STP_SELECT_INPUT for mode ALT7.

Table A-163. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS5

Offset: 0x0144 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-164. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_CS5. 000: Select mux mode: ALT0 mux port: NANDF_CS5 of instance: emi. 001: Select mux mode: ALT1 mux port: DA_1 of instance: pata. 010: Select mux mode: ALT2 mux port: TDATA[2] of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[21] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[0] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT2 of instance: esdhc4. 110: Select mux mode: ALT6 mux port: RST0 of instance: sim. 111: Select mux mode: ALT7 mux port: USBH3_DIR of instance: usboh3. NOTE: BGA contact NANDF_CS5 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DIR_SELECT_INPUT for mode ALT7.

Table A-165. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS6

Offset: 0x0148 (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-166. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS6 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS6. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_CS6. 000: Select mux mode: ALT0 mux port: NANDF_CS6 of instance: emi. 001: Select mux mode: ALT1 mux port: DA_2 of instance: pata. 010: Select mux mode: ALT2 mux port: TDATA[3] of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[22] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[1] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT3 of instance: esdhc4. 110: Select mux mode: ALT6 mux port: VEN0 of instance: sim. 111: Select mux mode: ALT7 mux port: SS3 of instance: cspi. NOTE: BGA contact NANDF_CS6 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_SS3_B_SELECT_INPUT for mode ALT7.

Table A-167. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_CS7

Offset: 0x014c (IOMUXC_SW_MUX_CTL_PAD_NANDF_CS7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-168. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_CS7 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_CS7. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: NANDF_CS7. 000: Select mux mode: ALT0 mux port: NANDF_CS7 of instance: emi. 001: Select mux mode: ALT1 mux port: TX_EN of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[23] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[2] of instance: sdma. 101: Select mux mode: ALT5 mux port: CLK of instance: esdhc3. 110: Select mux mode: ALT6 mux port: TX0 of instance: sim. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved.

Table A-169. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_RDY_INT

Offset: 0x0150 (IOMUXC_SW_MUX_CTL_PAD_NANDF_RDY_INT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-170. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_RDY_INT Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_RDY_INT. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: NANDF_RDY_INT. 000: Select mux mode: ALT0 mux port: RDY of instance: emi. 001: Select mux mode: ALT1 mux port: TX_CLK of instance: fec. 010: Select mux mode: ALT2 mux port: SS0 of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[24] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL[3] of instance: sdma. 101: Select mux mode: ALT5 mux port: CMD of instance: esdhc3. 110: Select mux mode: ALT6 mux port: RX0 of instance: sim. NOTE: BGA contact NANDF_RDY_INT is involved in Daisy Chain. - Config Register IOMUXC_EMI_IPP_IND_RDY_INT_SELECT_INPUT for mode ALT0. - Config Register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT for mode ALT1.

Table A-171. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D15

Offset: 0x0154 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-172. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D15 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D15. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D15. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[15] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[15] of instance: pata. 010: Select mux mode: ALT2 mux port: MOSI of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[25] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[0] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT7 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[0] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[0] of instance: gpu3d.

Table A-173. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D14

Offset: 0x0158 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D14) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-174. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D14 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D14. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D14. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[14] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[14] of instance: pata. 010: Select mux mode: ALT2 mux port: SS3 of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[26] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[1] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT6 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[1] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[1] of instance: gpu3d. NOTE: BGA contact NANDF_D14 is involved in Daisy Chain. - Config Register IOMUXC_ECSPI2_IPP_IND_SS_B_3_SELECT_INPUT for mode ALT2.

Table A-175. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D13

Offset: 0x015c (IOMUXC_SW_MUX_CTL_PAD_NANDF_D13) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-176. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D13 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D13. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D13. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[13] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[13] of instance: pata. 010: Select mux mode: ALT2 mux port: SS2 of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[27] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[2] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT5 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[2] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[2] of instance: gpu3d.

Table A-177. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D12

Offset	0x0160 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-178. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D12 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D12. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D12. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[12] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[12] of instance: pata. 010: Select mux mode: ALT2 mux port: SS1 of instance: ecspi2. 011: Select mux mode: ALT3 mux port: GPIO[28] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[3] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT4 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[3] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[3] of instance: gpu3d. NOTE: BGA contact NANDF_D12 is involved in Daisy Chain. - Config Register IOMUXC_ECSPI2_IPP_IND_SS_B_1_SELECT_INPUT for mode ALT2.

Table A-179. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D11

Offset: 0x0164 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-180. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D11 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D11. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D11. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[11] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[11] of instance: pata. 010: Select mux mode: ALT2 mux port: RX_DV of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[29] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[4] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT3 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[4] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[4] of instance: gpu3d. NOTE: BGA contact NANDF_D11 is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT3_IN_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT for mode ALT2.

Table A-181. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D10

Offset: 0x0168 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-182. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D10 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D10. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: NANDF_D10. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[10] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[10] of instance: pata. 011: Select mux mode: ALT3 mux port: GPIO[30] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[5] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT2 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[5] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[5] of instance: gpu3d. NOTE: BGA contact NANDF_D10 is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT2_IN_SELECT_INPUT for mode ALT5.

Table A-183. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D9

Offset: 0x016c (IOMUXC_SW_MUX_CTL_PAD_NANDF_D9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-184. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D9 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D9. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D9. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[9] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[9] of instance: pata. 010: Select mux mode: ALT2 mux port: RDATA[0] of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[31] of instance: gpio3. 100: Select mux mode: ALT4 mux port: DEBUG_PC[6] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT1 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[6] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[6] of instance: gpu3d. NOTE: BGA contact NANDF_D9 is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT1_IN_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_FEC_FEC_RDATA_0_SELECT_INPUT for mode ALT2.

Table A-185. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D8

Offset	0x0170 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D8)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-186. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D8 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D8. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D8. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[8] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[8] of instance: pata. 010: Select mux mode: ALT2 mux port: TDATA[0] of instance: fec. 011: Select mux mode: ALT3 mux port: GPIO[0] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[7] of instance: sdma. 101: Select mux mode: ALT5 mux port: DAT0 of instance: esdhc3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[7] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[7] of instance: gpu3d. NOTE: BGA contact NANDF_D8 is involved in Daisy Chain. - Config Register IOMUXC_ESDHC3_IPP_DAT0_IN_SELECT_INPUT for mode ALT5.

Table A-187. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D7

Offset: 0x0174 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-188. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D7 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D7. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: NANDF_D7. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[7] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[7] of instance: pata. 011: Select mux mode: ALT3 mux port: GPIO[1] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[8] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA0 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[8] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[8] of instance: gpu3d. NOTE: BGA contact NANDF_D7 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_0_SELECT_INPUT for mode ALT5.

Table A-189. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D6

Offset	0x0178 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-190. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D6 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D6. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D6. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[6] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[6] of instance: pata. 010: Select mux mode: ALT2 mux port: LCTL of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[2] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[9] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA1 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[9] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[9] of instance: gpu3d. NOTE: BGA contact NANDF_D6 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_1_SELECT_INPUT for mode ALT5.

Table A-191. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D5

Offset: 0x017c (IOMUXC_SW_MUX_CTL_PAD_NANDF_D5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-192. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D5. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[5] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[5] of instance: pata. 010: Select mux mode: ALT2 mux port: WP of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[3] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[10] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA2 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[10] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[10] of instance: gpu3d. NOTE: BGA contact NANDF_D5 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_2_SELECT_INPUT for mode ALT5.

Table A-193. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D4

Offset	0x0180 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-194. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D4. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[4] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[4] of instance: pata. 010: Select mux mode: ALT2 mux port: CD of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[4] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[11] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA3 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[11] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[11] of instance: gpu3d. NOTE: BGA contact NANDF_D4 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_3_SELECT_INPUT for mode ALT5.

Table A-195. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D3

Offset	0x0184 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-196. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D3. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[3] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[3] of instance: pata. 010: Select mux mode: ALT2 mux port: DAT4 of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[5] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[12] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA4 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[12] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[12] of instance: gpu3d. NOTE: BGA contact NANDF_D3 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_4_SELECT_INPUT for mode ALT5.

Table A-197. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D2

Offset	0x0188 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-198. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D2. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[2] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[2] of instance: pata. 010: Select mux mode: ALT2 mux port: DAT5 of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[6] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_PC[13] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA5 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[13] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[13] of instance: gpu3d. NOTE: BGA contact NANDF_D2 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_5_SELECT_INPUT for mode ALT5.

Table A-199. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D1

Offset: 0x018c (IOMUXC_SW_MUX_CTL_PAD_NANDF_D1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-200. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D1. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[1] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[1] of instance: pata. 010: Select mux mode: ALT2 mux port: DAT6 of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[7] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_CORE_STATE[0] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA6 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[14] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[14] of instance: gpu3d. NOTE: BGA contact NANDF_D1 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_6_SELECT_INPUT for mode ALT5.

Table A-201. Register: IOMUXC_SW_MUX_CTL_PAD_NANDF_D0

Offset	0x0190 (IOMUXC_SW_MUX_CTL_PAD_NANDF_D0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-202. Register IOMUXC_SW_MUX_CTL_PAD_NANDF_D0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact NANDF_D0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: NANDF_D0. 000: Select mux mode: ALT0 mux port: EIM_NFC_D[0] of instance: emi. 001: Select mux mode: ALT1 mux port: PATA_DATA[0] of instance: pata. 010: Select mux mode: ALT2 mux port: DAT7 of instance: esdhc4. 011: Select mux mode: ALT3 mux port: GPIO[8] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_CORE_STATE[1] of instance: sdma. 101: Select mux mode: ALT5 mux port: USBH3_DATA7 of instance: usboh3. 110: Select mux mode: ALT6 mux port: IPU_DIAG_BUS[15] of instance: ipu. 111: Select mux mode: ALT7 mux port: GPU_DEBUG_OUT[15] of instance: gpu3d. NOTE: BGA contact NANDF_D0 is involved in Daisy Chain. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_7_SELECT_INPUT for mode ALT5.

Table A-203. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D8

Offset: 0x0194 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-204. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D8 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D8. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: CSI1_D8. 00: Select mux mode: ALT0 mux port: CSI1_D[8] of instance: hsc_mipi_mix. 01: Select mux mode: ALT1 mux port: DETECT_Z of instance: hsc_mipi_mix. 11: Select mux mode: ALT3 mux port: GPIO[12] of instance: gpio3. NOTE: BGA contact CSI1_D8 is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_12_SELECT_INPUT for mode ALT3.

Table A-205. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D9

Offset: 0x0198 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-206. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D9 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D9. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: CSI1_D9. 00: Select mux mode: ALT0 mux port: CSI1_D[9] of instance: hsc_mipi_mix. 01: Select mux mode: ALT1 mux port: DELAY_Z of instance: hsc_mipi_mix. 11: Select mux mode: ALT3 mux port: GPIO[13] of instance: gpio3.

Table A-207. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D10

Offset: 0x019c (IOMUXC_SW_MUX_CTL_PAD_CSI1_D10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-208. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D10 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D10. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-209. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D11

Offset	0x01a0 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D11)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-210. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D11 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D11. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-211. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D12

Offset	0x01a4 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-212. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D12 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D12. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-213. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D13

Offset	0x01a8 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-214. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D13 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D13. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-215. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D14

Offset	0x01ac (IOMUXC_SW_MUX_CTL_PAD_CSI1_D14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-216. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D14 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D14. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-217. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D15

Offset	0x01b0 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-218. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D15 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D15. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-219. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D16

Offset	0x01b4 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-220. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D16 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D16. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-221. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D17

Offset	0x01b8 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-222. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D17 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D17. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-223. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D18

Offset	0x01bc (IOMUXC_SW_MUX_CTL_PAD_CSI1_D18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-224. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D18 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D18. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-225. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_D19

Offset	0x01c0 (IOMUXC_SW_MUX_CTL_PAD_CSI1_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-226. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_D19 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_D19. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-227. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_VSYNC

Offset: 0x01c4 (IOMUXC_SW_MUX_CTL_PAD_CSI1_VSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-228. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_VSYNC Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_VSYNC. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: CSI1_VSYNC. 00: Select mux mode: ALT0 mux port: CSI1_VSYNC of instance: hsc_mipi_mix. 01: Select mux mode: ALT1 mux port: TX_DDR_Q of instance: mipi_dphy_trippi_1_slave. 11: Select mux mode: ALT3 mux port: GPIO[14] of instance: gpio3.

Table A-229. Register: IOMUXC_SW_MUX_CTL_PAD_CSI1_HSYNC

Offset: 0x01c8 (IOMUXC_SW_MUX_CTL_PAD_CSI1_HSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-230. Register IOMUXC_SW_MUX_CTL_PAD_CSI1_HSYNC Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI1_HSYNC. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: CSI1_HSYNC. 00: Select mux mode: ALT0 mux port: CSI1_HSYNC of instance: hsc_mipi_mix. 01: Select mux mode: ALT1 mux port: TX_DDR_I of instance: mipi_dphy_trippi_1_slave. 11: Select mux mode: ALT3 mux port: GPIO[15] of instance: gpio3.

Table A-231. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D12

Offset: 0x01cc (IOMUXC_SW_MUX_CTL_PAD_CSI2_D12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-232. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D12 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D12. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: CSI2_D12. 00: Select mux mode: ALT0 mux port: CSI2_D[12] of instance: hsc_mipi_mix. 01: Select mux mode: ALT1 mux port: LP_RX_E of instance: hsc_mipi_mix. 11: Select mux mode: ALT3 mux port: GPIO[9] of instance: gpio4.

Table A-233. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D13

Offset	0x01d0 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-234. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D13 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D13. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSI2_D13. 000: Select mux mode: ALT0 mux port: CSI2_D[13] of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: RX_VALID_ESC_OUT of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: GPIO[10] of instance: gpio4. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[45] of instance: emi.

Table A-235. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D14

Offset	0x01d4 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-236. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D14 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D14. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-237. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D15

Offset	0x01d8 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-238. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D15 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D15. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-239. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D16

Offset	0x01dc (IOMUXC_SW_MUX_CTL_PAD_CSI2_D16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-240. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D16 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D16. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-241. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D17

Offset: 0x01e0 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D17) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-242. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D17 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D17. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-243. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D18

Offset: 0x01e4 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D18) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-243. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D18

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-244. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D18 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D18. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: CSI2_D18. 000: Select mux mode: ALT0 mux port: CSI2_D[18] of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: HS_TX_E of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: OBSRV_INT_OUT0 of instance: elvis_observe_mux. 011: Select mux mode: ALT3 mux port: GPIO[11] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_RTBUFFER_WRITE of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[46] of instance: emi.

Table A-245. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D19

Offset	0x01e8 (IOMUXC_SW_MUX_CTL_PAD_CSI2_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table A-245. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_D19

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-246. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_D19 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_D19. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: CSI2_D19. 000: Select mux mode: ALT0 mux port: CSI2_D[19] of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: LP_TX_E of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: OBSRV_INT_OUT1 of instance: elvis_observe_mux. 011: Select mux mode: ALT3 mux port: GPIO[12] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_YIELD of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[47] of instance: emi.

Table A-247. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_VSYNC

Offset 0x01ec (IOMUXC_SW_MUX_CTL_PAD_CSI2_VSYNC) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-248. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_VSYNC Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_VSYNC. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: CSI2_VSYNC. 000: Select mux mode: ALT0 mux port: CSI2_VSYNC of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: HS_RX_E of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: OBSRV_INT_OUT2 of instance: elvis_observe_mux. 011: Select mux mode: ALT3 mux port: GPIO[13] of instance: gpio4. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_ERROR of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[48] of instance: emi.

Table A-249. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_HSYNC

Offset: 0x01f0 (IOMUXC_SW_MUX_CTL_PAD_CSI2_HSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-250. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_HSYNC Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_HSYNC. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-250. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_HSYNC Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: CSI2_HSYNC. 000: Select mux mode: ALT0 mux port: CSI2_HSYNC of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: TX_BYTE_CLK_HS_OUT of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: OBSRV_INT_OUT3 of instance: elvis_observe_mux. 011: Select mux mode: ALT3 mux port: GPIO[14] of instance: gpio4. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[49] of instance: emi.

Table A-251. Register: IOMUXC_SW_MUX_CTL_PAD_CSI2_PIXCLK

Offset: 0x01f4 (IOMUXC_SW_MUX_CTL_PAD_CSI2_PIXCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-252. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_PIXCLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSI2_PIXCLK. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-252. Register IOMUXC_SW_MUX_CTL_PAD_CSI2_PIXCLK Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: CSI2_PIXCLK. 000: Select mux mode: ALT0 mux port: CSI2_PIXCLK of instance: hsc_mipi_mix. 001: Select mux mode: ALT1 mux port: RX_BYTE_CLK_HS_OUT of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: OBSRV_INT_OUT4 of instance: elvis_observe_mux. 011: Select mux mode: ALT3 mux port: GPIO[15] of instance: gpio4. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[50] of instance: emi.

Table A-253. Register: IOMUXC_SW_MUX_CTL_PAD_I2C1_CLK

Offset 0x01f8 (IOMUXC_SW_MUX_CTL_PAD_I2C1_CLK) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-254. Register IOMUXC_SW_MUX_CTL_PAD_I2C1_CLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact I2C1_CLK. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: I2C1_CLK. 00: Select mux mode: ALT0 mux port: SCL of instance: hsi2c. 11: Select mux mode: ALT3 mux port: GPIO[16] of instance: gpio4.

Table A-255. Register: IOMUXC_SW_MUX_CTL_PAD_I2C1_DAT

Offset	0x01fc (IOMUXC_SW_MUX_CTL_PAD_I2C1_DAT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0		0	0		
W												SION			MUX_MODE	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-256. Register IOMUXC_SW_MUX_CTL_PAD_I2C1_DAT Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact I2C1_DAT. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: I2C1_DAT. 00: Select mux mode: ALT0 mux port: SDA of instance: hsi2c. 11: Select mux mode: ALT3 mux port: GPIO[17] of instance: gpio4.

Table A-257. Register: IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_TXD

Offset	0x0200 (IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_TXD)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table A-257. Register: IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_TXD

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-258. Register IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_TXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact AUD3_BB_TXD. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: AUD3_BB_TXD. 000: Select mux mode: ALT0 mux port: AUD3_TXD of instance: audmux. 001: Select mux mode: ALT1 mux port: DATA of instance: slm. 011: Select mux mode: ALT3 mux port: GPIO[18] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[0] of instance: usbphy.

Table A-259. Register: IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_RXD

Offset: 0x0204 (IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_RXD)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-260. Register IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_RXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact AUD3_BB_RXD. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: AUD3_BB_RXD. 000: Select mux mode: ALT0 mux port: AUD3_RXD of instance: audmux. 001: Select mux mode: ALT1 mux port: RXD_MUX of instance: uart3. 011: Select mux mode: ALT3 mux port: GPIO[19] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[1] of instance: usbphy. NOTE: BGA contact AUD3_BB_RXD is involved in Daisy Chain. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-261. Register: IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_CK

Offset: 0x0208 (IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_CK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-262. Register IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_CK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact AUD3_BB_CK. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-262. Register IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_CK Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: AUD3_BB_CK. 000: Select mux mode: ALT0 mux port: AUD3_TXC of instance: audmux. 001: Select mux mode: ALT1 mux port: CLK of instance: slm. 011: Select mux mode: ALT3 mux port: GPIO[20] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[2] of instance: usbphy.

Table A-263. Register: IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_FS

Offset: 0x020c (IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_FS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-264. Register IOMUXC_SW_MUX_CTL_PAD_AUD3_BB_FS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact AUD3_BB_FS. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: AUD3_BB_FS. 000: Select mux mode: ALT0 mux port: AUD3_TXFS of instance: audmux. 001: Select mux mode: ALT1 mux port: TXD_MUX of instance: uart3. 011: Select mux mode: ALT3 mux port: GPIO[21] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[3] of instance: usbphy. NOTE: BGA contact AUD3_BB_FS is involved in Daisy Chain. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-265. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_MOSI

Offset	0x0210 (IOMUXC_SW_MUX_CTL_PAD_CSPI1_MOSI)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-266. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_MOSI Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_MOSI. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_MOSI. 000: Select mux mode: ALT0 mux port: MOSI of instance: ecspi1. 001: Select mux mode: ALT1 mux port: SDA of instance: i2c1. 011: Select mux mode: ALT3 mux port: GPIO[22] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[4] of instance: usbphy. NOTE: BGA contact CSPI1_MOSI is involved in Daisy Chain. - Config Register IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT for mode ALT1.

Table A-267. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_MISO

Offset	0x0214 (IOMUXC_SW_MUX_CTL_PAD_CSPI1_MISO)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-267. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_MISO

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-268. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_MISO Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_MISO. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_MISO. 000: Select mux mode: ALT0 mux port: MISO of instance: ecspi1. 001: Select mux mode: ALT1 mux port: AUD4_RXD of instance: audmux. 011: Select mux mode: ALT3 mux port: GPIO[23] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[5] of instance: usbphy. NOTE: BGA contact CSPI1_MISO is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_DA_AMX_SELECT_INPUT for mode ALT1.

Table A-269. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS0

Offset: 0x0218 (IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS0)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-270. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_SS0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_SS0. 000: Select mux mode: ALT0 mux port: SS0 of instance: ecspi1. 001: Select mux mode: ALT1 mux port: AUD4_TXC of instance: audmux. 011: Select mux mode: ALT3 mux port: GPIO[24] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[6] of instance: usbphy. NOTE: BGA contact CSPI1_SS0 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1.

Table A-271. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS1

Offset: 0x021c (IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-272. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_SS1. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-272. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_SS1 Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_SS1. 000: Select mux mode: ALT0 mux port: SS1 of instance: ecspi1. 001: Select mux mode: ALT1 mux port: AUD4_TXD of instance: audmux. 011: Select mux mode: ALT3 mux port: GPIO[25] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[7] of instance: usbphy. NOTE: BGA contact CSPI1_SS1 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_DB_AMX_SELECT_INPUT for mode ALT1.

Table A-273. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_RDY

Offset: 0x0220 (IOMUXC_SW_MUX_CTL_PAD_CSPI1_RDY) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-274. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_RDY Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_RDY. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-274. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_RDY Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_RDY. 000: Select mux mode: ALT0 mux port: RDY of instance: ecspi1. 001: Select mux mode: ALT1 mux port: AUD4_TXFS of instance: audmux. 011: Select mux mode: ALT3 mux port: GPIO[26] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[8] of instance: usbphy. NOTE: BGA contact CSPI1_RDY is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P4_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1.

Table A-275. Register: IOMUXC_SW_MUX_CTL_PAD_CSPI1_SCLK

Offset 0x0224 (IOMUXC_SW_MUX_CTL_PAD_CSPI1_SCLK) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-276. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_SCLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact CSPI1_SCLK. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-276. Register IOMUXC_SW_MUX_CTL_PAD_CSPI1_SCLK Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: CSPI1_SCLK. 000: Select mux mode: ALT0 mux port: SCLK of instance: ecspi1. 001: Select mux mode: ALT1 mux port: SCL of instance: i2c1. 011: Select mux mode: ALT3 mux port: GPIO[27] of instance: gpio4. 111: Select mux mode: ALT7 mux port: DATAOUT[9] of instance: usbphy. NOTE: BGA contact CSPI1_SCLK is involved in Daisy Chain. - Config Register IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT for mode ALT1.

Table A-277. Register: IOMUXC_SW_MUX_CTL_PAD_UART1_RXD

Offset: 0x0228 (IOMUXC_SW_MUX_CTL_PAD_UART1_RXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-278. Register IOMUXC_SW_MUX_CTL_PAD_UART1_RXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART1_RXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-278. Register IOMUXC_SW_MUX_CTL_PAD_UART1_RXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: UART1_RXD. 000: Select mux mode: ALT0 mux port: RXD_MUX of instance: uart1. 011: Select mux mode: ALT3 mux port: GPIO[28] of instance: gpio4. 101: Select mux mode: ALT5 mux port: READY_ESC_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[12] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[10] of instance: usbphy. NOTE: BGA contact UART1_RXD is involved in Daisy Chain. - Config Register IOMUXC_UART1_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT0.

Table A-279. Register: IOMUXC_SW_MUX_CTL_PAD_UART1_TXD

Offset: 0x022c (IOMUXC_SW_MUX_CTL_PAD_UART1_TXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-280. Register IOMUXC_SW_MUX_CTL_PAD_UART1_TXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART1_TXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-280. Register IOMUXC_SW_MUX_CTL_PAD_UART1_TXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: UART1_TXD. 000: Select mux mode: ALT0 mux port: TXD_MUX of instance: uart1. 001: Select mux mode: ALT1 mux port: PWMO of instance: pwm2. 011: Select mux mode: ALT3 mux port: GPIO[29] of instance: gpio4. 101: Select mux mode: ALT5 mux port: REQUEST_ESC_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[13] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[11] of instance: usbphy. NOTE: BGA contact UART1_TXD is involved in Daisy Chain. - Config Register IOMUXC_UART1_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT0.

Table A-281. Register: IOMUXC_SW_MUX_CTL_PAD_UART1_RTS

Offset 0x0230 (IOMUXC_SW_MUX_CTL_PAD_UART1_RTS) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-282. Register IOMUXC_SW_MUX_CTL_PAD_UART1_RTS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART1_RTS. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-282. Register IOMUXC_SW_MUX_CTL_PAD_UART1_RTS Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: UART1_RTS. 000: Select mux mode: ALT0 mux port: RTS of instance: uart1. 001: Select mux mode: ALT1 mux port: Reserved of instance: Reserved. 011: Select mux mode: ALT3 mux port: GPIO[30] of instance: gpio4. 101: Select mux mode: ALT5 mux port: CLK_ESC_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[14] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[12] of instance: usbphy. NOTE: BGA contact UART1_RTS is involved in Daisy Chain. - Config Register IOMUXC_UART1_IPP_UART_RTS_B_SELECT_INPUT for mode ALT0.

Table A-283. Register: IOMUXC_SW_MUX_CTL_PAD_UART1_CTS

Offset 0x0234 (IOMUXC_SW_MUX_CTL_PAD_UART1_CTS) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-284. Register IOMUXC_SW_MUX_CTL_PAD_UART1_CTS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART1_CTS. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-284. Register IOMUXC_SW_MUX_CTL_PAD_UART1_CTS Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: UART1_CTS. 000: Select mux mode: ALT0 mux port: CTS of instance: uart1. 001: Select mux mode: ALT1 mux port: Reserved of instance: Reserved. 011: Select mux mode: ALT3 mux port: GPIO[31] of instance: gpio4. 101: Select mux mode: ALT5 mux port: READY_HS_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[15] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[13] of instance: usbphy. NOTE: BGA contact UART1_CTS is involved in Daisy Chain. - Config Register IOMUXC_UART1_IPP_UART_RTS_B_SELECT_INPUT for mode ALT0.

Table A-285. Register: IOMUXC_SW_MUX_CTL_PAD_UART2_RXD

Offset 0x0238 (IOMUXC_SW_MUX_CTL_PAD_UART2_RXD) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-286. Register IOMUXC_SW_MUX_CTL_PAD_UART2_RXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART2_RXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-286. Register IOMUXC_SW_MUX_CTL_PAD_UART2_RXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: UART2_RXD. 000: Select mux mode: ALT0 mux port: RXD_MUX of instance: uart2. 001: Select mux mode: ALT1 mux port: TXD of instance: firi. 011: Select mux mode: ALT3 mux port: GPIO[20] of instance: gpio1. 101: Select mux mode: ALT5 mux port: VALID_HS_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[16] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[14] of instance: usbphy. NOTE: BGA contact UART2_RXD is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT0.

Table A-287. Register: IOMUXC_SW_MUX_CTL_PAD_UART2_TXD

Offset 0x023c (IOMUXC_SW_MUX_CTL_PAD_UART2_TXD) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-288. Register IOMUXC_SW_MUX_CTL_PAD_UART2_TXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART2_TXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-288. Register IOMUXC_SW_MUX_CTL_PAD_UART2_TXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: UART2_TXD. 000: Select mux mode: ALT0 mux port: TXD_MUX of instance: uart2. 001: Select mux mode: ALT1 mux port: RXD of instance: firi. 011: Select mux mode: ALT3 mux port: GPIO[21] of instance: gpio1. 101: Select mux mode: ALT5 mux port: VALID_ESC_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[17] of instance: emi. 111: Select mux mode: ALT7 mux port: DATAOUT[15] of instance: usbphy. NOTE: BGA contact UART2_TXD is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT0.

Table A-289. Register: IOMUXC_SW_MUX_CTL_PAD_UART3_RXD

Offset: 0x0240 (IOMUXC_SW_MUX_CTL_PAD_UART3_RXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-290. Register IOMUXC_SW_MUX_CTL_PAD_UART3_RXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART3_RXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-290. Register IOMUXC_SW_MUX_CTL_PAD_UART3_RXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: UART3_RXD. 000: Select mux mode: ALT0 mux port: DTR of instance: uart1. 001: Select mux mode: ALT1 mux port: RXD_MUX of instance: uart3. 010: Select mux mode: ALT2 mux port: CSI1_D[0] of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: GPIO[22] of instance: gpio1. 101: Select mux mode: ALT5 mux port: SYNC_HS_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[18] of instance: emi. 111: Select mux mode: ALT7 mux port: LINESTATE[0] of instance: usbphy. NOTE: BGA contact UART3_RXD is involved in Daisy Chain. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-291. Register: IOMUXC_SW_MUX_CTL_PAD_UART3_TXD

Offset: 0x0244 (IOMUXC_SW_MUX_CTL_PAD_UART3_TXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-292. Register IOMUXC_SW_MUX_CTL_PAD_UART3_TXD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact UART3_TXD. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-292. Register IOMUXC_SW_MUX_CTL_PAD_UART3_TXD Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: UART3_TXD. 000: Select mux mode: ALT0 mux port: DSR of instance: uart1. 001: Select mux mode: ALT1 mux port: TXD_MUX of instance: uart3. 010: Select mux mode: ALT2 mux port: CSI1_D[1] of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: GPIO[23] of instance: gpio1. 101: Select mux mode: ALT5 mux port: REQUEST_HS_OUT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[19] of instance: emi. 111: Select mux mode: ALT7 mux port: LINESTATE[1] of instance: usbphy. NOTE: BGA contact UART3_TXD is involved in Daisy Chain. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-293. Register: IOMUXC_SW_MUX_CTL_PAD_OWIRE_LINE

Offset: 0x0248 (IOMUXC_SW_MUX_CTL_PAD_OWIRE_LINE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Table A-294. Register IOMUXC_SW_MUX_CTL_PAD_OWIRE_LINE Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact OWIRE_LINE. 0: Input Path is determined by functionality of the selected mux mode (regular).

Table A-294. Register IOMUXC_SW_MUX_CTL_PAD_OWIRE_LINE Bits Description

Field	Description
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: OWIRE_LINE. 000: Select mux mode: ALT0 mux port: LINE of instance: owire. 011: Select mux mode: ALT3 mux port: GPIO[24] of instance: gpio1. 101: Select mux mode: ALT5 mux port: CTI_TRIGOUT6 of instance: tigerp_platform_ne_32k_256k. 110: Select mux mode: ALT6 mux port: OUT1 of instance: spdif. 111: Select mux mode: ALT7 mux port: SYSTEM_RST of instance: src.

Table A-295. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0

Offset: 0x024c (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-296. Register IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: KEY_ROW0. 000: Select mux mode: ALT0 mux port: ROW[0] of instance: kpp. 001: Select mux mode: ALT1 mux port: DSTROBE of instance: emi. 010: Select mux mode: ALT2 mux port: TX_DDR_Q of instance: mipi_dphy_trippi_2_master. 101: Select mux mode: ALT5 mux port: RANDOM_V of instance: scc. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[20] of instance: emi.

Table A-297. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1

Offset	0x0250 (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-298. Register IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: KEY_ROW1. 000: Select mux mode: ALT0 mux port: ROW[1] of instance: kpp. 001: Select mux mode: ALT1 mux port: RTIC_SEC_VIO of instance: rtic. 010: Select mux mode: ALT2 mux port: TX_DDR_I of instance: mipi_dphy_trippi_2_master. 101: Select mux mode: ALT5 mux port: RANDOM of instance: scc. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[21] of instance: emi.

Table A-299. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2

Offset	0x0254 (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table A-299. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2

R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-300. Register IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: KEY_ROW2. 000: Select mux mode: ALT0 mux port: ROW[2] of instance: kpp. 001: Select mux mode: ALT1 mux port: RTIC_DONE_INT of instance: rtic. 010: Select mux mode: ALT2 mux port: TX_DDR_Q of instance: mipi_dphy_trippi_1_master. 101: Select mux mode: ALT5 mux port: DONE of instance: sjc. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[22] of instance: emi.

Table A-301. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3

Offset	0x0258 (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-302. Register IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: KEY_ROW3. 000: Select mux mode: ALT0 mux port: ROW[3] of instance: kpp. 001: Select mux mode: ALT1 mux port: CSU_ALARM_AUT[0] of instance: csu. 010: Select mux mode: ALT2 mux port: TX_DDR_I of instance: mipi_dphy_trippi_1_master. 101: Select mux mode: ALT5 mux port: FAIL of instance: sjc. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[23] of instance: emi.

Table A-303. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL0

Offset 0x025c (IOMUXC_SW_MUX_CTL_PAD_KEY_COL0) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-304. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL0 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: KEY_COL0. 000: Select mux mode: ALT0 mux port: COL[0] of instance: kpp. 001: Select mux mode: ALT1 mux port: CSU_ALARM_AUT[1] of instance: csu. 010: Select mux mode: ALT2 mux port: HS_TX_E0 of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: PLL1_BYP of instance: ccm. NOTE: BGA contact KEY_COL0 is involved in Daisy Chain. - Config Register IOMUXC_CCM_PLL1_BYPASS_CLK_SELECT_INPUT for mode ALT7.

Table A-305. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL1

Offset	0x0260 (IOMUXC_SW_MUX_CTL_PAD_KEY_COL1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-306. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL1 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: KEY_COL1. 000: Select mux mode: ALT0 mux port: COL[1] of instance: kpp. 001: Select mux mode: ALT1 mux port: CSU_ALARM_AUT[2] of instance: csu. 010: Select mux mode: ALT2 mux port: HS_TX_E1 of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: PLL2_BYP of instance: ccm. NOTE: BGA contact KEY_COL1 is involved in Daisy Chain. - Config Register IOMUXC_CCM_PLL2_BYPASS_CLK_SELECT_INPUT for mode ALT7.

Table A-307. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL2

Offset	0x0264 (IOMUXC_SW_MUX_CTL_PAD_KEY_COL2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Table A-307. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL2

R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-308. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL2 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: KEY_COL2. 000: Select mux mode: ALT0 mux port: COL[2] of instance: kpp. 001: Select mux mode: ALT1 mux port: SNOOP2 of instance: ipu. 111: Select mux mode: ALT7 mux port: PLL3_BYP of instance: ccm.

Table A-309. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL3

Offset	0x0268 (IOMUXC_SW_MUX_CTL_PAD_KEY_COL3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-310. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL3 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: KEY_COL3. 000: Select mux mode: ALT0 mux port: COL[3] of instance: kpp. 101: Select mux mode: ALT5 mux port: CTI_TRIGOUT7 of instance: tigerp_platform_ne_32k_256k. 111: Select mux mode: ALT7 mux port: INT_BOOT of instance: src.

Table A-311. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL4

Offset	0x026c (IOMUXC_SW_MUX_CTL_PAD_KEY_COL4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-312. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact KEY_COL4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: KEY_COL4. 000: Select mux mode: ALT0 mux port: COL[4] of instance: kpp. 001: Select mux mode: ALT1 mux port: RI of instance: uart1. 010: Select mux mode: ALT2 mux port: RTS of instance: uart3. 011: Select mux mode: ALT3 mux port: SCL of instance: i2c2. 100: Select mux mode: ALT4 mux port: SSI_EXT2_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: CTI_TRIGIN_ACK7 of instance: tigerp_platform_ne_32k_256k. 110: Select mux mode: ALT6 mux port: OUT1 of instance: spdif. 111: Select mux mode: ALT7 mux port: ANY_PU_RST of instance: src. NOTE: BGA contact KEY_COL4 is involved in Daisy Chain. - Config Register IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT for mode ALT3. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT2.

Table A-313. Register: IOMUXC_SW_MUX_CTL_PAD_KEY_COL5

Offset: 0x0270 (IOMUXC_SW_MUX_CTL_PAD_KEY_COL5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-314. Register IOMUXC_SW_MUX_CTL_PAD_KEY_COL5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact KEY_COL5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: KEY_COL5. 000: Select mux mode: ALT0 mux port: COL[5] of instance: kpp. 001: Select mux mode: ALT1 mux port: DCD of instance: uart1. 010: Select mux mode: ALT2 mux port: CTS of instance: uart3. 011: Select mux mode: ALT3 mux port: SDA of instance: i2c2. 100: Select mux mode: ALT4 mux port: SSI_EXT1_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: CTI_TRIGIN7 of instance: tigerp_platform_ne_32k_256k. 110: Select mux mode: ALT6 mux port: MCT_EXT_ACT_TRIG of instance: hsc_mipi_mix. 111: Select mux mode: ALT7 mux port: JTAG_ACT of instance: sjc. NOTE: BGA contact KEY_COL5 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_PAR_SISG_TRIG_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT for mode ALT3. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT2.

Table A-315. Register: IOMUXC_SW_MUX_CTL_PAD_JTAG_DE_B

Offset: 0x0274 (IOMUXC_SW_MUX_CTL_PAD_JTAG_DE_B)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-316. Register IOMUXC_SW_MUX_CTL_PAD_JTAG_DE_B Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact JTAG_DE_B. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-317. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_CLK

Offset: 0x0278 (IOMUXC_SW_MUX_CTL_PAD_USBH1_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-318. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_CLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_CLK. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: USBH1_CLK. 000: Select mux mode: ALT0 mux port: USBH1_CLK of instance: usboh3. 001: Select mux mode: ALT1 mux port: SCLK of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[25] of instance: gpio1. 011: Select mux mode: ALT3 mux port: LPDT_ESC_OUT of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_CORE_RUN of instance: sdma. 101: Select mux mode: ALT5 mux port: SCL of instance: i2c2. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[0] of instance: emi. 111: Select mux mode: ALT7 mux port: CTS of instance: uart3. NOTE: BGA contact USBH1_CLK is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT7.

Table A-319. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DIR

Offset	0x027c (IOMUXC_SW_MUX_CTL_PAD_USBH1_DIR)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-320. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DIR Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DIR. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: USBH1_DIR. 000: Select mux mode: ALT0 mux port: USBH1_DIR of instance: usboh3. 001: Select mux mode: ALT1 mux port: MOSI of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[26] of instance: gpio1. 011: Select mux mode: ALT3 mux port: HS_RX_Z of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_MODE of instance: sdma. 101: Select mux mode: ALT5 mux port: SDA of instance: i2c2. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[1] of instance: emi. 111: Select mux mode: ALT7 mux port: RTS of instance: uart3. NOTE: BGA contact USBH1_DIR is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT for mode ALT7.

Table A-321. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_STP

Offset: 0x0280 (IOMUXC_SW_MUX_CTL_PAD_USBH1_STP) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-322. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_STP Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_STP. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: USBH1_STP. 000: Select mux mode: ALT0 mux port: USBH1_STP of instance: usboh3. 001: Select mux mode: ALT1 mux port: RDY of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[27] of instance: gpio1. 011: Select mux mode: ALT3 mux port: ACTIVE_HS_OUT of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_EVENT_CHANNEL_SEL of instance: sdma. 101: Select mux mode: ALT5 mux port: RXD_MUX of instance: uart3. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[2] of instance: emi. 111: Select mux mode: ALT7 mux port: BISTOK of instance: usbphy. NOTE: BGA contact USBH1_STP is involved in Daisy Chain. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT5.

Table A-323. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_NXT

Offset: 0x0284 (IOMUXC_SW_MUX_CTL_PAD_USBH1_NXT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-324. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_NXT Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_NXT. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: USBH1_NXT. 000: Select mux mode: ALT0 mux port: USBH1_NXT of instance: usboh3. 001: Select mux mode: ALT1 mux port: MISO of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[28] of instance: gpio1. 011: Select mux mode: ALT3 mux port: RX_TERM_E of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_RWB of instance: sdma. 101: Select mux mode: ALT5 mux port: TXD_MUX of instance: uart3. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[3] of instance: emi. 111: Select mux mode: ALT7 mux port: ONBIST of instance: usbphy. NOTE: BGA contact USBH1_NXT is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT5.

Table A-325. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA0

Offset: 0x0288 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-326. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA0. 000: Select mux mode: ALT0 mux port: USBH1_DATA0 of instance: usboh3. 001: Select mux mode: ALT1 mux port: CTS of instance: uart2. 010: Select mux mode: ALT2 mux port: GPIO[11] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[0] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_CORE_STATE[2] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[4] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[0] of instance: usbphy. NOTE: BGA contact USBH1_DATA0 is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT1.

Table A-327. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA1

Offset: 0x028c (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-328. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA1. 000: Select mux mode: ALT0 mux port: USBH1_DATA1 of instance: usboh3. 001: Select mux mode: ALT1 mux port: RXD_MUX of instance: uart2. 010: Select mux mode: ALT2 mux port: GPIO[12] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[1] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_CORE_STATE[3] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[5] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[1] of instance: usbphy. NOTE: BGA contact USBH1_DATA1 is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-329. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA2

Offset: 0x0290 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-330. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA2. 000: Select mux mode: ALT0 mux port: USBH1_DATA2 of instance: usboh3. 001: Select mux mode: ALT1 mux port: TXD_MUX of instance: uart2. 010: Select mux mode: ALT2 mux port: GPIO[13] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[2] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_MATCHED_DMBUS of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[6] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[2] of instance: usbphy. NOTE: BGA contact USBH1_DATA2 is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT1.

Table A-331. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA3

Offset: 0x0294 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-332. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA3. 000: Select mux mode: ALT0 mux port: USBH1_DATA3 of instance: usboh3. 001: Select mux mode: ALT1 mux port: RTS of instance: uart2. 010: Select mux mode: ALT2 mux port: GPIO[14] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[3] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_DEVICE[0] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[7] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[3] of instance: usbphy. NOTE: BGA contact USBH1_DATA3 is involved in Daisy Chain. - Config Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT for mode ALT1.

Table A-333. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA4

Offset: 0x0298 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-334. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA4. 000: Select mux mode: ALT0 mux port: USBH1_DATA4 of instance: usboh3. 001: Select mux mode: ALT1 mux port: SS0 of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[15] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[4] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_DEVICE[1] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[8] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[4] of instance: usbphy.

Table A-335. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA5

Offset: 0x029c (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-336. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA5. 000: Select mux mode: ALT0 mux port: USBH1_DATA5 of instance: usboh3. 001: Select mux mode: ALT1 mux port: SS1 of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[16] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[5] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_DEVICE[2] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[9] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[5] of instance: usbphy. NOTE: BGA contact USBH1_DATA5 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_SS1_B_SELECT_INPUT for mode ALT1.

Table A-337. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA6

Offset: 0x02a0 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-338. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA6 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA6. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: USBH1_DATA6. 000: Select mux mode: ALT0 mux port: USBH1_DATA6 of instance: usboh3. 001: Select mux mode: ALT1 mux port: SS3 of instance: cspi. 010: Select mux mode: ALT2 mux port: GPIO[17] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[6] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_DEVICE[3] of instance: sdma. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[10] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[6] of instance: usbphy. NOTE: BGA contact USBH1_DATA6 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_SS3_B_SELECT_INPUT for mode ALT1.

Table A-339. Register: IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA7

Offset: 0x02a4 (IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Table A-340. Register IOMUXC_SW_MUX_CTL_PAD_USBH1_DATA7 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact USBH1_DATA7. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: USBH1_DATA7. 000: Select mux mode: ALT0 mux port: USBH1_DATA7 of instance: usboh3. 001: Select mux mode: ALT1 mux port: SS3 of instance: ecspi1. 010: Select mux mode: ALT2 mux port: GPIO[18] of instance: gpio1. 011: Select mux mode: ALT3 mux port: DATA_ESC_OUT[7] of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DEBUG_BUS_DEVICE[4] of instance: sdma. 101: Select mux mode: ALT5 mux port: SS3 of instance: ecspi2. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[11] of instance: emi. 111: Select mux mode: ALT7 mux port: VSTATUS[7] of instance: usbphy. NOTE: BGA contact USBH1_DATA7 is involved in Daisy Chain. - Config Register IOMUXC_ECSPI2_IPP_IND_SS_B_3_SELECT_INPUT for mode ALT5.

Table A-341. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_PIN11

Offset: 0x02a8 (IOMUXC_SW_MUX_CTL_PAD_DI1_PIN11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-342. Register IOMUXC_SW_MUX_CTL_PAD_DI1_PIN11 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DI1_PIN11. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DI1_PIN11. 000: Select mux mode: ALT0 mux port: DI1_PIN11 of instance: ipu. 001: Select mux mode: ALT1 mux port: READY_ESC_IN of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: GPIO[0] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. 111: Select mux mode: ALT7 mux port: SS2 of instance: ecspi1.

Table A-343. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_PIN12

Offset: 0x02ac (IOMUXC_SW_MUX_CTL_PAD_DI1_PIN12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-344. Register IOMUXC_SW_MUX_CTL_PAD_DI1_PIN12 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DI1_PIN12. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI1_PIN12. 000: Select mux mode: ALT0 mux port: DI1_PIN12 of instance: ipu. 001: Select mux mode: ALT1 mux port: REQUEST_ESC_IN of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: GPIO[1] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI1_PIN12 is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_1_SELECT_INPUT for mode ALT4.

Table A-345. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_PIN13

Offset: 0x02b0 (IOMUXC_SW_MUX_CTL_PAD_DI1_PIN13) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-346. Register IOMUXC_SW_MUX_CTL_PAD_DI1_PIN13 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DI1_PIN13. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI1_PIN13. 000: Select mux mode: ALT0 mux port: DI1_PIN13 of instance: ipu. 001: Select mux mode: ALT1 mux port: CLK_ESC_IN of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: GPIO[2] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI1_PIN13 is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_2_SELECT_INPUT for mode ALT4.

Table A-347. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_D0_CS

Offset	0x02b4 (IOMUXC_SW_MUX_CTL_PAD_DI1_D0_CS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-348. Register IOMUXC_SW_MUX_CTL_PAD_DI1_D0_CS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DI1_D0_CS. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI1_D0_CS. 000: Select mux mode: ALT0 mux port: DI1_D0_CS of instance: ipu. 001: Select mux mode: ALT1 mux port: LPDT_ESC_IN of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: GPIO[3] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI1_D0_CS is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_3_SELECT_INPUT for mode ALT4.

Table A-349. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_D1_CS

Offset: 0x02b8 (IOMUXC_SW_MUX_CTL_PAD_DI1_D1_CS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-350. Register IOMUXC_SW_MUX_CTL_PAD_DI1_D1_CS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DI1_D1_CS. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: DI1_D1_CS. 000: Select mux mode: ALT0 mux port: DI1_D1_CS of instance: ipu. 001: Select mux mode: ALT1 mux port: READY_HS_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI1_PIN14 of instance: ipu. 011: Select mux mode: ALT3 mux port: DI1_PIN5 of instance: ipu. 100: Select mux mode: ALT4 mux port: GPIO[4] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI1_D1_CS is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_4_SELECT_INPUT for mode ALT4.

Table A-351. Register: IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIN

Offset: 0x02bc (IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIN) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-352. Register IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIN Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISPB2_SER_DIN. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DISPB2_SER_DIN. 000: Select mux mode: ALT0 mux port: DISPB2_SER_DIN of instance: ipu. 001: Select mux mode: ALT1 mux port: VALID_HS_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI1_PIN1 of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: GPIO[5] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISPB2_SER_DIN is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_5_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_HSC_MIPI_MIX_PAR0_VSYNC_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_IPU_IPP_DI_1_IND_DISPB_SD_D_SELECT_INPUT for mode ALT0.

Table A-353. Register: IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIO

Offset: 0x02c0 (IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIO) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-354. Register IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_DIO Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISPB2_SER_DIO. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: DISPB2_SER_DIO. 000: Select mux mode: ALT0 mux port: DISPB2_SER_DIO of instance: ipu. 001: Select mux mode: ALT1 mux port: ACTIVE_HS_IN of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: DI1_PIN6 of instance: ipu. 100: Select mux mode: ALT4 mux port: GPIO[6] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. 110: Select mux mode: ALT6 mux port: WDOG_RST_B_DEB of instance: wdog1. NOTE: BGA contact DISPB2_SER_DIO is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_6_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_IPU_IPP_DI_1_IND_DISPB_SD_D_SELECT_INPUT for mode ALT0.

Table A-355. Register: IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_CLK

Offset: 0x02c4 (IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-356. Register IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_CLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISPB2_SER_CLK. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DISPB2_SER_CLK. 000: Select mux mode: ALT0 mux port: DISPB2_SER_CLK of instance: ipu. 001: Select mux mode: ALT1 mux port: BYTE_CLK_HS_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI1_PIN17 of instance: ipu. 011: Select mux mode: ALT3 mux port: DI1_PIN7 of instance: ipu. 100: Select mux mode: ALT4 mux port: GPIO[7] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. 110: Select mux mode: ALT6 mux port: WDOG_RST_B_DEB of instance: wdog2. NOTE: BGA contact DISPB2_SER_CLK is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_7_SELECT_INPUT for mode ALT4.

Table A-357. Register: IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_RS

Offset: 0x02c8 (IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_RS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-358. Register IOMUXC_SW_MUX_CTL_PAD_DISPB2_SER_RS Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISPB2_SER_RS. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: DISPB2_SER_RS. 000: Select mux mode: ALT0 mux port: DISPB2_SER_RS of instance: ipu. 001: Select mux mode: ALT1 mux port: VALID_ESC_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI1_PIN16 of instance: ipu. 011: Select mux mode: ALT3 mux port: DI1_PIN8 of instance: ipu. 100: Select mux mode: ALT4 mux port: GPIO[8] of instance: gpio3. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISPB2_SER_RS is involved in Daisy Chain. - Config Register IOMUXC_GPIO3_IPP_IND_G_IN_8_SELECT_INPUT for mode ALT4.

Table A-359. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT0

Offset: 0x02cc (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-360. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-361. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT1

Offset: 0x02d0 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-362. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-363. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT2

Offset	0x02d4 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-364. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-365. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT3

Offset	0x02d8 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-366. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-367. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT4

Offset: 0x02dc (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-368. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-369. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT5

Offset	0x02e0 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-370. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP1_DAT5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-371. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT6

Offset	0x02e4 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-372. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT6 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT6. 000: Select mux mode: ALT0 mux port: DISP1_DAT[6] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_USB_SRC of instance: src.

Table A-373. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT7

Offset: 0x02e8 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-374. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT7 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT7. 000: Select mux mode: ALT0 mux port: DISP1_DAT[7] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_EEPROM_CFG of instance: src.

Table A-375. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT8

Offset	0x02ec (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT8)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-376. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT8 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT8. 000: Select mux mode: ALT0 mux port: DISP1_DAT[8] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_SRC[0] of instance: src.

Table A-377. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT9

Offset: 0x02f0 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-378. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT9 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT9. 000: Select mux mode: ALT0 mux port: DISP1_DAT[9] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_SRC[1] of instance: src.

Table A-379. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT10

Offset	0x02f4 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT10)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-380. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT10 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT10. 000: Select mux mode: ALT0 mux port: DISP1_DAT[10] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_SPARE_SIZE of instance: src.

Table A-381. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT11

Offset	0x02f8 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT11)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-382. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT11 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT11. 000: Select mux mode: ALT0 mux port: DISP1_DAT[11] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_LPB_FREQ[2] of instance: src.

Table A-383. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT12

Offset	0x02fc (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-384. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT12 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT12. 000: Select mux mode: ALT0 mux port: DISP1_DAT[12] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_MLC_SEL of instance: src.

Table A-385. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT13

Offset	0x0300 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-386. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT13 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT13. 000: Select mux mode: ALT0 mux port: DISP1_DAT[13] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_MEM_CTL[0] of instance: src.

Table A-387. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT14

Offset	0x0304 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-388. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT14 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT14. 000: Select mux mode: ALT0 mux port: DISP1_DAT[14] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_MEM_CTL[1] of instance: src.

Table A-389. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT15

Offset: 0x0308 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-390. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT15 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT15. 000: Select mux mode: ALT0 mux port: DISP1_DAT[15] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_BUS_WIDTH of instance: src.

Table A-391. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT16

Offset	0x030c (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-392. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT16 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT16. 000: Select mux mode: ALT0 mux port: DISP1_DAT[16] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_PAGE_SIZE[0] of instance: src.

Table A-393. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT17

Offset	0x0310 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-394. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT17 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DISP1_DAT17. 000: Select mux mode: ALT0 mux port: DISP1_DAT[17] of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_PAGE_SIZE[1] of instance: src.

Table A-395. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT18

Offset	0x0314 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-396. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT18 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DISP1_DAT18. 000: Select mux mode: ALT0 mux port: DISP1_DAT[18] of instance: ipu. 100: Select mux mode: ALT4 mux port: DI2_PIN5 of instance: ipu. 101: Select mux mode: ALT5 mux port: DI2_PIN11 of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_WEIM_MUXED[0] of instance: src.

Table A-397. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT19

Offset	0x0318 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-398. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT19 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DISP1_DAT19. 000: Select mux mode: ALT0 mux port: DISP1_DAT[19] of instance: ipu. 100: Select mux mode: ALT4 mux port: DI2_PIN6 of instance: ipu. 101: Select mux mode: ALT5 mux port: DI2_PIN12 of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_WEIM_MUXED[1] of instance: src.

Table A-399. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT20

Offset	0x031c (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT20)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-400. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT20 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DISP1_DAT20. 000: Select mux mode: ALT0 mux port: DISP1_DAT[20] of instance: ipu. 100: Select mux mode: ALT4 mux port: DI2_PIN7 of instance: ipu. 101: Select mux mode: ALT5 mux port: DI2_PIN13 of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_MEM_TYPE[0] of instance: src.

Table A-401. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT21

Offset	0x0320 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT21)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-402. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT21 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DISP1_DAT21. 000: Select mux mode: ALT0 mux port: DISP1_DAT[21] of instance: ipu. 100: Select mux mode: ALT4 mux port: DI2_PIN8 of instance: ipu. 101: Select mux mode: ALT5 mux port: DI2_PIN14 of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_MEM_TYPE[1] of instance: src.

Table A-403. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT22

Offset: 0x0324 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT22) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-404. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT22 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DISP1_DAT22. 000: Select mux mode: ALT0 mux port: DISP1_DAT[22] of instance: ipu. 011: Select mux mode: ALT3 mux port: CTI_TRIGOUT_ACK6 of instance: tigerp_platform_ne_32k_256k. 101: Select mux mode: ALT5 mux port: DISP2_DAT[16] of instance: ipu. 110: Select mux mode: ALT6 mux port: DI2_D0_CS of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_LPB_FREQ[0] of instance: src.

Table A-405. Register: IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT23

Offset	0x0328 (IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT23)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-406. Register IOMUXC_SW_MUX_CTL_PAD_DISP1_DAT23 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: DISP1_DAT23. 000: Select mux mode: ALT0 mux port: DISP1_DAT[23] of instance: ipu. 011: Select mux mode: ALT3 mux port: CTI_TRIGOUT_ACK7 of instance: tigerp_platform_ne_32k_256k. 100: Select mux mode: ALT4 mux port: SER_DISP2_CS of instance: ipu. 101: Select mux mode: ALT5 mux port: DISP2_DAT[17] of instance: ipu. 110: Select mux mode: ALT6 mux port: DI2_D1_CS of instance: ipu. 111: Select mux mode: ALT7 mux port: BT_LPB_FREQ[1] of instance: src.

Table A-407. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_PIN3

Offset	0x032c (IOMUXC_SW_MUX_CTL_PAD_DI1_PIN3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-408. Register IOMUXC_SW_MUX_CTL_PAD_DI1_PIN3 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI1_PIN3. 000: Select mux mode: ALT0 mux port: DI1_PIN3 of instance: ipu. 001: Select mux mode: ALT1 mux port: TX_DDR_Q of instance: mipi_dphy_trippi_4_slave. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[24] of instance: emi. 111: Select mux mode: ALT7 mux port: SIM_TX_CLK_TEST of instance: sim.

Table A-409. Register: IOMUXC_SW_MUX_CTL_PAD_DI1_PIN2

Offset	0x0330 (IOMUXC_SW_MUX_CTL_PAD_DI1_PIN2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-410. Register IOMUXC_SW_MUX_CTL_PAD_DI1_PIN2 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI1_PIN2. 000: Select mux mode: ALT0 mux port: DI1_PIN2 of instance: ipu. 001: Select mux mode: ALT1 mux port: TX_DDR_I of instance: mipi_dphy_trippi_4_slave. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[25] of instance: emi. 111: Select mux mode: ALT7 mux port: SIM_RCV_CLK_TEST of instance: sim.

Table A-411. Register: IOMUXC_SW_MUX_CTL_PAD_DI_GP1

Offset	0x0334 (IOMUXC_SW_MUX_CTL_PAD_DI_GP1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-412. Register IOMUXC_SW_MUX_CTL_PAD_DI_GP1 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DI_GP1. 000: Select mux mode: ALT0 mux port: DISPB1_SER_RS of instance: ipu. 001: Select mux mode: ALT1 mux port: RX_VALID_ESC_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI1_EXT_CLK of instance: ccm. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[26] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI_GP1 is involved in Daisy Chain. - Config Register IOMUXC_CCM_IPP_DI0_CLK_SELECT_INPUT for mode ALT2.

Table A-413. Register: IOMUXC_SW_MUX_CTL_PAD_DI_GP2

Offset	0x0338 (IOMUXC_SW_MUX_CTL_PAD_DI_GP2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-414. Register IOMUXC_SW_MUX_CTL_PAD_DI_GP2 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DI_GP2. 000: Select mux mode: ALT0 mux port: DISPB1_SER_CLK of instance: ipu. 001: Select mux mode: ALT1 mux port: SYNC_HS_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: DI2_WAIT of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[27] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI_GP2 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_PAR1_DI_WAIT_SELECT_INPUT for mode ALT2.

Table A-415. Register: IOMUXC_SW_MUX_CTL_PAD_DI_GP3

Offset	0x033c (IOMUXC_SW_MUX_CTL_PAD_DI_GP3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-416. Register IOMUXC_SW_MUX_CTL_PAD_DI_GP3 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: DI_GP3. 000: Select mux mode: ALT0 mux port: DISPB1_SER_DIO of instance: ipu. 001: Select mux mode: ALT1 mux port: REQUEST_HS_IN of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TX_ER of instance: fec. 011: Select mux mode: ALT3 mux port: CSI1_DATA_EN of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[28] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI_GP3 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS2_DATA_EN_SELECT_INPUT for mode ALT3. - Config Register IOMUXC_IPU_IPP_DI_0_IND_DISPB_SD_D_SELECT_INPUT for mode ALT0.

Table A-417. Register: IOMUXC_SW_MUX_CTL_PAD_DI2_PIN4

Offset	0x0340 (IOMUXC_SW_MUX_CTL_PAD_DI2_PIN4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-418. Register IOMUXC_SW_MUX_CTL_PAD_DI2_PIN4 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 5 iomux modes to be used for BGA contact: DI2_PIN4. 000: Select mux mode: ALT0 mux port: DI2_PIN4 of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[0] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: CRS of instance: fec. 011: Select mux mode: ALT3 mux port: CSI2_DATA_EN of instance: hsc_mipi_mix. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[29] of instance: emi. NOTE: BGA contact DI2_PIN4 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_CRS_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT for mode ALT3.

Table A-419. Register: IOMUXC_SW_MUX_CTL_PAD_DI2_PIN2

Offset	0x0344 (IOMUXC_SW_MUX_CTL_PAD_DI2_PIN2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-420. Register IOMUXC_SW_MUX_CTL_PAD_DI2_PIN2 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI2_PIN2. 000: Select mux mode: ALT0 mux port: DI2_PIN2 of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[1] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: MDC of instance: fec. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[30] of instance: emi.

Table A-421. Register: IOMUXC_SW_MUX_CTL_PAD_DI2_PIN3

Offset	0x0348 (IOMUXC_SW_MUX_CTL_PAD_DI2_PIN3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-422. Register IOMUXC_SW_MUX_CTL_PAD_DI2_PIN3 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: DI2_PIN3. 000: Select mux mode: ALT0 mux port: DI2_PIN3 of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[2] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: MDIO of instance: fec. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[31] of instance: emi. NOTE: BGA contact DI2_PIN3 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_MDI_SELECT_INPUT for mode ALT2.

Table A-423. Register: IOMUXC_SW_MUX_CTL_PAD_DI2_DISP_CLK

Offset: 0x034c (IOMUXC_SW_MUX_CTL_PAD_DI2_DISP_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-424. Register IOMUXC_SW_MUX_CTL_PAD_DI2_DISP_CLK Bits Description

Field	Description
31-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: DI2_DISP_CLK. 00: Select mux mode: ALT0 mux port: DI2_DISP_CLK of instance: ipu. 10: Select mux mode: ALT2 mux port: RDATA[1] of instance: fec. NOTE: BGA contact DI2_DISP_CLK is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_RDATA_1_SELECT_INPUT for mode ALT2.

Table A-425. Register: IOMUXC_SW_MUX_CTL_PAD_DI_GP4

Offset	0x0350 (IOMUXC_SW_MUX_CTL_PAD_DI_GP4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-426. Register IOMUXC_SW_MUX_CTL_PAD_DI_GP4 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DI_GP4. 000: Select mux mode: ALT0 mux port: DISPB1_SER_DIN of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[3] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RDATA[2] of instance: fec. 011: Select mux mode: ALT3 mux port: DI2_PIN1 of instance: hsc_mipi_mix. 100: Select mux mode: ALT4 mux port: DI2_PIN15 of instance: ipu. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[32] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DI_GP4 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_RDATA_2_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_IPU_IPP_DI_0_IND_DISPB_SD_D_SELECT_INPUT for mode ALT0.

Table A-427. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT0

Offset	0x0354 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-428. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT0 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT0. 000: Select mux mode: ALT0 mux port: DISP2_DAT[0] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[4] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RDATA[3] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_CLK of instance: usboh3. 100: Select mux mode: ALT4 mux port: COL[6] of instance: kpp. 101: Select mux mode: ALT5 mux port: RXD_MUX of instance: uart3. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[33] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT0 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_RDATA_3_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_KPP_IPP_IND_COL_6_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_CLK_SELECT_INPUT for mode ALT3.

Table A-429. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT1

Offset: 0x0358 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-430. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT1 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT1. 000: Select mux mode: ALT0 mux port: DISP2_DAT[1] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[5] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RX_ER of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DIR of instance: usboh3. 100: Select mux mode: ALT4 mux port: COL[7] of instance: kpp. 101: Select mux mode: ALT5 mux port: TXD_MUX of instance: uart3. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[34] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT1 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_KPP_IPP_IND_COL_7_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT for mode ALT5. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DIR_SELECT_INPUT for mode ALT3.

Table A-431. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT2

Offset	0x035c (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-432. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-433. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT3

Offset: 0x0360 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-434. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-435. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT4

Offset	0x0364 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-436. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-437. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT5

Offset	0x0368 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-438. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-0	Reserved

Table A-439. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT6

Offset	0x036c (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-440. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT6 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT6. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT6. 000: Select mux mode: ALT0 mux port: DISP2_DAT[6] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[6] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TDATA[1] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_STP of instance: usboh3. 100: Select mux mode: ALT4 mux port: ROW[4] of instance: kpp. 101: Select mux mode: ALT5 mux port: GPIO[19] of instance: gpio1. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[35] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT6 is involved in Daisy Chain. - Config Register IOMUXC_KPP_IPP_IND_ROW_4_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_STP_SELECT_INPUT for mode ALT3.

Table A-441. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT7

Offset: 0x0370 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-442. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT7 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT7. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT7. 000: Select mux mode: ALT0 mux port: DISP2_DAT[7] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_HS_IN[7] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TDATA[2] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_NXT of instance: usboh3. 100: Select mux mode: ALT4 mux port: ROW[5] of instance: kpp. 101: Select mux mode: ALT5 mux port: GPIO[29] of instance: gpio1. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[36] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT7 is involved in Daisy Chain. - Config Register IOMUXC_KPP_IPP_IND_ROW_5_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_NXT_SELECT_INPUT for mode ALT3.

Table A-443. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT8

Offset: 0x0374 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-444. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT8 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT8. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT8. 000: Select mux mode: ALT0 mux port: DISP2_DAT[8] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[0] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TDATA[3] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA0 of instance: usboh3. 100: Select mux mode: ALT4 mux port: ROW[6] of instance: kpp. 101: Select mux mode: ALT5 mux port: GPIO[30] of instance: gpio1. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[37] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT8 is involved in Daisy Chain. - Config Register IOMUXC_KPP_IPP_IND_ROW_6_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_0_SELECT_INPUT for mode ALT3.

Table A-445. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT9

Offset: 0x0378 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-446. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT9 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT9. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT9. 000: Select mux mode: ALT0 mux port: DISP2_DAT[9] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[1] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TX_EN of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA1 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_RXC of instance: audmux. 101: Select mux mode: ALT5 mux port: GPIO[31] of instance: gpio1. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[38] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT9 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_1_SELECT_INPUT for mode ALT3.

Table A-447. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT10

Offset: 0x037c (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-448. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT10 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT10. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT10. 000: Select mux mode: ALT0 mux port: DISP2_DAT[10] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[2] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: COL of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA2 of instance: usboh3. 100: Select mux mode: ALT4 mux port: ROW[7] of instance: kpp. 101: Select mux mode: ALT5 mux port: SER_DISP2_CS of instance: ipu. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[39] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT10 is involved in Daisy Chain. - Config Register IOMUXC_FEC_FEC_COL_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_KPP_IPP_IND_ROW_7_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_2_SELECT_INPUT for mode ALT3.

Table A-449. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT11

Offset	0x0380 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT11)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-450. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT11 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact DISP2_DAT11. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DISP2_DAT11. 000: Select mux mode: ALT0 mux port: DISP2_DAT[11] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[3] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RX_CLK of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA3 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_TXD of instance: audmux. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[40] of instance: emi. 111: Select mux mode: ALT7 mux port: GPIO[10] of instance: gpio1. NOTE: BGA contact DISP2_DAT11 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_DB_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_3_SELECT_INPUT for mode ALT3.

Table A-451. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT12

Offset	0x0384 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-452. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT12 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DISP2_DAT12. 000: Select mux mode: ALT0 mux port: DISP2_DAT[12] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[4] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RX_DV of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA4 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_RXD of instance: audmux. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[41] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT12 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_DA_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_4_SELECT_INPUT for mode ALT3.

Table A-453. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT13

Offset	0x0388 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-454. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT13 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DISP2_DAT13. 000: Select mux mode: ALT0 mux port: DISP2_DAT[13] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[5] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TX_CLK of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA5 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_TXC of instance: audmux. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[42] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT13 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_5_SELECT_INPUT for mode ALT3.

Table A-455. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT14

Offset	0x038c (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-456. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT14 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: DISP2_DAT14. 000: Select mux mode: ALT0 mux port: DISP2_DAT[14] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[6] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: RDATA[0] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA6 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_TXFS of instance: audmux. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[43] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT14 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_FEC_FEC_RDATA_0_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_6_SELECT_INPUT for mode ALT3.

Table A-457. Register: IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT15

Offset: 0x0390 (IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-458. Register IOMUXC_SW_MUX_CTL_PAD_DISP2_DAT15 Bits Description

Field	Description
31-3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: DISP2_DAT15. 000: Select mux mode: ALT0 mux port: DISP2_DAT[15] of instance: ipu. 001: Select mux mode: ALT1 mux port: DATA_ESC_IN[7] of instance: hsc_mipi_mix. 010: Select mux mode: ALT2 mux port: TDATA[0] of instance: fec. 011: Select mux mode: ALT3 mux port: USBH3_DATA7 of instance: usboh3. 100: Select mux mode: ALT4 mux port: AUD6_RXFS of instance: audmux. 101: Select mux mode: ALT5 mux port: SER_DISP1_CS of instance: ipu. 110: Select mux mode: ALT6 mux port: EMI_DEBUG[44] of instance: emi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact DISP2_DAT15 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P6_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_7_SELECT_INPUT for mode ALT3.

Table A-459. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_CMD

Offset: 0x0394 (IOMUXC_SW_MUX_CTL_PAD_SD1_CMD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-460. Register IOMUXC_SW_MUX_CTL_PAD_SD1_CMD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD1_CMD. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD1_CMD. 00: Select mux mode: ALT0 mux port: CMD of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_RXFS of instance: audmux. 10: Select mux mode: ALT2 mux port: MOSI of instance: cspi. NOTE: BGA contact SD1_CMD is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT for mode ALT2.

Table A-461. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_CLK

Offset: 0x0398 (IOMUXC_SW_MUX_CTL_PAD_SD1_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-462. Register IOMUXC_SW_MUX_CTL_PAD_SD1_CLK Bits Description

Field	Description
31-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD1_CLK. 00: Select mux mode: ALT0 mux port: CLK of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_RXC of instance: audmux. 10: Select mux mode: ALT2 mux port: SCLK of instance: cspi. NOTE: BGA contact SD1_CLK is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT for mode ALT2.

Table A-463. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0

Offset: 0x039c (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-464. Register IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD1_DATA0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD1_DATA0. 00: Select mux mode: ALT0 mux port: DAT0 of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_TXD of instance: audmux. 10: Select mux mode: ALT2 mux port: MISO of instance: cspi. NOTE: BGA contact SD1_DATA0 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT for mode ALT2.

Table A-465. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1

Offset: 0x03a0 (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-466. Register IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD1_DATA1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: SD1_DATA1. 00: Select mux mode: ALT0 mux port: DAT1 of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_RXD of instance: audmux. NOTE: BGA contact SD1_DATA1 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT for mode ALT1.

Table A-467. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2

Offset: 0x03a4 (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-468. Register IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD1_DATA2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: SD1_DATA2. 00: Select mux mode: ALT0 mux port: DAT2 of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_TXC of instance: audmux. NOTE: BGA contact SD1_DATA2 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1.

Table A-469. Register: IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3

Offset: 0x03a8 (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-470. Register IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD1_DATA3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD1_DATA3. 00: Select mux mode: ALT0 mux port: DAT3 of instance: esdhc1. 01: Select mux mode: ALT1 mux port: AUD5_TXFS of instance: audmux. 10: Select mux mode: ALT2 mux port: SS1 of instance: cspi. NOTE: BGA contact SD1_DATA3 is involved in Daisy Chain. - Config Register IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. - Config Register IOMUXC_CSPI_IPP_IND_SS1_B_SELECT_INPUT for mode ALT2.

Table A-471. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_0

Offset	0x03ac (IOMUXC_SW_MUX_CTL_PAD_GPIO1_0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-472. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: GPIO1_0. 000: Select mux mode: ALT0 mux port: CD of instance: esdhc1. 001: Select mux mode: ALT1 mux port: GPIO[0] of instance: gpio1. 010: Select mux mode: ALT2 mux port: SS2 of instance: cspi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact GPIO1_0 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_SS2_B_SELECT_INPUT for mode ALT2.

Table A-473. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_1

Offset: 0x03b0 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-474. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 4 iomux modes to be used for BGA contact: GPIO1_1. 000: Select mux mode: ALT0 mux port: WP of instance: esdhc1. 001: Select mux mode: ALT1 mux port: GPIO[1] of instance: gpio1. 010: Select mux mode: ALT2 mux port: MISO of instance: cspi. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved. NOTE: BGA contact GPIO1_1 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT for mode ALT2.

Table A-475. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_CMD

Offset: 0x03b4 (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-476. Register IOMUXC_SW_MUX_CTL_PAD_SD2_CMD Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_CMD. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_CMD. 00: Select mux mode: ALT0 mux port: CMD of instance: esdhc2. 01: Select mux mode: ALT1 mux port: SCL of instance: i2c1. 10: Select mux mode: ALT2 mux port: MOSI of instance: cspi. NOTE: BGA contact SD2_CMD is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT for mode ALT1.

Table A-477. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_CLK

Offset: 0x03b8 (IOMUXC_SW_MUX_CTL_PAD_SD2_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-478. Register IOMUXC_SW_MUX_CTL_PAD_SD2_CLK Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_CLK. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_CLK. 00: Select mux mode: ALT0 mux port: CLK of instance: esdhc2. 01: Select mux mode: ALT1 mux port: SDA of instance: i2c1. 10: Select mux mode: ALT2 mux port: SCLK of instance: cspi. NOTE: BGA contact SD2_CLK is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT for mode ALT2. - Config Register IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT for mode ALT1.

Table A-479. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0

Offset: 0x03bc (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-480. Register IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_DATA0. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_DATA0. 00: Select mux mode: ALT0 mux port: DAT0 of instance: esdhc2. 01: Select mux mode: ALT1 mux port: DAT4 of instance: esdhc1. 10: Select mux mode: ALT2 mux port: MISO of instance: cspi. NOTE: BGA contact SD2_DATA0 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT for mode ALT2.

Table A-481. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1

Offset: 0x03c0 (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-482. Register IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_DATA1. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_DATA1. 00: Select mux mode: ALT0 mux port: DAT1 of instance: esdhc2. 01: Select mux mode: ALT1 mux port: DAT5 of instance: esdhc1. 10: Select mux mode: ALT2 mux port: H2_DP of instance: usboh3.

Table A-483. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2

Offset: 0x03c4 (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-484. Register IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_DATA2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_DATA2. 00: Select mux mode: ALT0 mux port: DAT2 of instance: esdhc2. 01: Select mux mode: ALT1 mux port: DAT6 of instance: esdhc1. 10: Select mux mode: ALT2 mux port: H2_DM of instance: usboh3.

Table A-485. Register: IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3

Offset: 0x03c8 (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-486. Register IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact SD2_DATA3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3-2	Reserved
1-0 MUX_MODE	MUX Mode Select Field. Select 1 of 3 iomux modes to be used for BGA contact: SD2_DATA3. 00: Select mux mode: ALT0 mux port: DAT3 of instance: esdhc2. 01: Select mux mode: ALT1 mux port: DAT7 of instance: esdhc1. 10: Select mux mode: ALT2 mux port: SS2 of instance: cspi. NOTE: BGA contact SD2_DATA3 is involved in Daisy Chain. - Config Register IOMUXC_CSPI_IPP_IND_SS2_B_SELECT_INPUT for mode ALT2.

Table A-487. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_2

Offset	0x03cc (IOMUXC_SW_MUX_CTL_PAD_GPIO1_2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-488. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_2 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_2. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: GPIO1_2. 000: Select mux mode: ALT0 mux port: GPIO[2] of instance: gpio1. 001: Select mux mode: ALT1 mux port: PWMO of instance: pwm1. 010: Select mux mode: ALT2 mux port: SCL of instance: i2c2. 101: Select mux mode: ALT5 mux port: CCM_OUT_2 of instance: ccm. 110: Select mux mode: ALT6 mux port: TOG_EN of instance: dpllip1. 111: Select mux mode: ALT7 mux port: PLL1_BYP of instance: ccm. NOTE: BGA contact GPIO1_2 is involved in Daisy Chain. - Config Register IOMUXC_CCM_PLL1_BYPASS_CLK_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT for mode ALT6. - Config Register IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT for mode ALT2.

Table A-489. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_3

Offset	0x03d0 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-490. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_3 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_3. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 6 iomux modes to be used for BGA contact: GPIO1_3. 000: Select mux mode: ALT0 mux port: GPIO[3] of instance: gpio1. 001: Select mux mode: ALT1 mux port: PWMO of instance: pwm2. 010: Select mux mode: ALT2 mux port: SDA of instance: i2c2. 101: Select mux mode: ALT5 mux port: CLKO2 of instance: ccm. 110: Select mux mode: ALT6 mux port: CLKIN of instance: gpt. 111: Select mux mode: ALT7 mux port: PLL2_BYP of instance: ccm. NOTE: BGA contact GPIO1_3 is involved in Daisy Chain. - Config Register IOMUXC_CCM_PLL2_BYPASS_CLK_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT for mode ALT2.

Table A-491. Register: IOMUXC_SW_MUX_CTL_PAD_PMIC_INT_REQ

Offset: 0x03d4 (IOMUXC_SW_MUX_CTL_PAD_PMIC_INT_REQ) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	MUX_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-492. Register IOMUXC_SW_MUX_CTL_PAD_PMIC_INT_REQ Bits Description

Field	Description
31-1	Reserved
0 MUX_MODE	MUX Mode Select Field. Select 1 of 2 iomux modes to be used for BGA contact: PMIC_INT_REQ. 00: Select mux mode: ALT0 mux port: PWRFAIL_INT of instance: tzic. 01: Select mux mode: ALT1 mux port: PMU_IRQ_B of instance: tigerp_platform_ne_32k_256k.

Table A-493. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_4

Offset	0x03d8 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-494. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_4 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_4. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: GPIO1_4. 000: Select mux mode: ALT0 mux port: GPIO[4] of instance: gpio1. 001: Select mux mode: ALT1 mux port: SDMA_EXT_EVENT[0] of instance: sdma. 010: Select mux mode: ALT2 mux port: WDOG_B of instance: wdog1. 011: Select mux mode: ALT3 mux port: RDY of instance: emi. 100: Select mux mode: ALT4 mux port: DI2_EXT_CLK of instance: ccm. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. 110: Select mux mode: ALT6 mux port: CAPIN1 of instance: gpt. 111: Select mux mode: ALT7 mux port: TOG_EN of instance: dpllip1. NOTE: BGA contact GPIO1_4 is involved in Daisy Chain. - Config Register IOMUXC_CCM_IPP_DI1_CLK_SELECT_INPUT for mode ALT4. - Config Register IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT for mode ALT7. - Config Register IOMUXC_EMI_IPP_IND_RDY_INT_SELECT_INPUT for mode ALT3.

Table A-495. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_5

Offset	0x03dc (IOMUXC_SW_MUX_CTL_PAD_GPIO1_5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-496. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_5 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_5. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: GPIO1_5. 000: Select mux mode: ALT0 mux port: GPIO[5] of instance: gpio1. 001: Select mux mode: ALT1 mux port: SDMA_EXT_EVENT[1] of instance: sdma. 010: Select mux mode: ALT2 mux port: WDOG_B of instance: wdog2. 011: Select mux mode: ALT3 mux port: DI2_PIN16 of instance: ipu. 101: Select mux mode: ALT5 mux port: CLKO of instance: ccm. 110: Select mux mode: ALT6 mux port: CSI2_MCLK of instance: ccm. 111: Select mux mode: ALT7 mux port: Reserved of instance: Reserved.

Table A-497. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_6

Offset	0x03e0 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-498. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_6 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_6. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: GPIO1_6. 000: Select mux mode: ALT0 mux port: GPIO[6] of instance: gpio1. 001: Select mux mode: ALT1 mux port: SSI_EXT2_CLK of instance: ccm. 010: Select mux mode: ALT2 mux port: MCT_EXT_ACT_TRIG of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: REF_EN_B of instance: ccm. 100: Select mux mode: ALT4 mux port: DI2_PIN17 of instance: ipu. 101: Select mux mode: ALT5 mux port: EPITO of instance: epit2. 110: Select mux mode: ALT6 mux port: CAPIN2 of instance: gpt. 111: Select mux mode: ALT7 mux port: TD of instance: csu. NOTE: BGA contact GPIO1_6 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_PAR_SISG_TRIG_SELECT_INPUT for mode ALT2.

Table A-499. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_7

Offset	0x03e4 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_7)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-500. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_7 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_7. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: GPIO1_7. 000: Select mux mode: ALT0 mux port: GPIO[7] of instance: gpio1. 001: Select mux mode: ALT1 mux port: SSI_EXT1_CLK of instance: ccm. 010: Select mux mode: ALT2 mux port: OUT1 of instance: spdif. 011: Select mux mode: ALT3 mux port: CCM_OUT_0 of instance: ccm. 101: Select mux mode: ALT5 mux port: EPITO of instance: epit1. 110: Select mux mode: ALT6 mux port: WP of instance: esdhc2. 111: Select mux mode: ALT7 mux port: TOG_EN of instance: dpllip1. NOTE: BGA contact GPIO1_7 is involved in Daisy Chain. - Config Register IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT for mode ALT7.

Table A-501. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_8

Offset	0x03e8 (IOMUXC_SW_MUX_CTL_PAD_GPIO1_8)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-502. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_8 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_8. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for BGA contact: GPIO1_8. 000: Select mux mode: ALT0 mux port: GPIO[8] of instance: gpio1. 001: Select mux mode: ALT1 mux port: USB_PWR of instance: usboh3. 010: Select mux mode: ALT2 mux port: CSI2_DATA_EN of instance: hsc_mipi_mix. 011: Select mux mode: ALT3 mux port: Reserved of instance: Reserved. 100: Select mux mode: ALT4 mux port: CLKO2 of instance: ccm. 110: Select mux mode: ALT6 mux port: CD of instance: esdhc2. 111: Select mux mode: ALT7 mux port: TESTER_ACK of instance: src. NOTE: BGA contact GPIO1_8 is involved in Daisy Chain. - Config Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT for mode ALT2.

Table A-503. Register: IOMUXC_SW_MUX_CTL_PAD_GPIO1_9

Offset: 0x03ec (IOMUXC_SW_MUX_CTL_PAD_GPIO1_9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-504. Register IOMUXC_SW_MUX_CTL_PAD_GPIO1_9 Bits Description

Field	Description
31-5	Reserved
4 SION	Software Input On Field. Force the selected mux mode Input path no matter of MUX_MODE functionality. 1: Force input path of BGA contact GPIO1_9. 0: Input Path is determined by functionality of the selected mux mode (regular).
3	Reserved
2-0 MUX_MODE	MUX Mode Select Field. Select 1 of 8 iomux modes to be used for BGA contact: GPIO1_9. 000: Select mux mode: ALT0 mux port: GPIO[9] of instance: gpio1. 001: Select mux mode: ALT1 mux port: USB_OC of instance: usboh3. 010: Select mux mode: ALT2 mux port: DI2_D1_CS of instance: ipu. 011: Select mux mode: ALT3 mux port: CCM_OUT_1 of instance: ccm. 100: Select mux mode: ALT4 mux port: CLKO of instance: ccm. 101: Select mux mode: ALT5 mux port: Reserved of instance: Reserved. 110: Select mux mode: ALT6 mux port: LCTL of instance: esdhc2. 111: Select mux mode: ALT7 mux port: SER_DISP2_CS of instance: ipu.

Table A-505. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D16

Offset: 0x03f0 (IOMUXC_SW_PAD_CTL_PAD_EIM_D16) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-506. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D16 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D16. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D16. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D16. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D16. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-506. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D16 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D16. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D16. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D16. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-507. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D17

Offset	0x03f4 (IOMUXC_SW_PAD_CTL_PAD_EIM_D17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-508. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D17 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D17. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D17. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D17. 0: Keeper 1: Pull
5-3	Reserved

Table A-508. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D17 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D17. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D17. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-509. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D18

Offset	0x03f8 (IOMUXC_SW_PAD_CTL_PAD_EIM_D18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-510. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D18 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D18. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D18. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D18. 0: Keeper 1: Pull
5-3	Reserved

Table A-510. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D18 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D18. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D18. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-511. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D19

Offset	0x03fc (IOMUXC_SW_PAD_CTL_PAD_EIM_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-512. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D19 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D19. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D19. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D19. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D19. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-512. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D19 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D19. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D19. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D19. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-513. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D20

Offset	0x0400 (IOMUXC_SW_PAD_CTL_PAD_EIM_D20)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-514. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D20 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D20. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D20. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D20. 0: Keeper 1: Pull
5-3	Reserved

Table A-514. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D20 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D20. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D20. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-515. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D21

Offset	0x0404 (IOMUXC_SW_PAD_CTL_PAD_EIM_D21)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-516. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D21 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D21. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D21. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D21. 0: Keeper 1: Pull
5-3	Reserved

Table A-516. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D21 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D21. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D21. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-517. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D22

Offset	0x0408 (IOMUXC_SW_PAD_CTL_PAD_EIM_D22)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-518. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D22 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D22. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D22. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D22. 0: Keeper 1: Pull
5-3	Reserved

Table A-518. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D22 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D22. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D22. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-519. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D23

Offset	0x040c (IOMUXC_SW_PAD_CTL_PAD_EIM_D23)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-520. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D23 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D23. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D23. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D23. 0: Keeper 1: Pull
5-3	Reserved

Table A-520. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D23 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D23. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D23. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-521. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D24

Offset	0x0410 (IOMUXC_SW_PAD_CTL_PAD_EIM_D24)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-522. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D24 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D24. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D24. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D24. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D24. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-522. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D24 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D24. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D24. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D24. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-523. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D25

Offset	0x0414 (IOMUXC_SW_PAD_CTL_PAD_EIM_D25)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-524. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D25 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D25. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D25. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D25. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D25. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-524. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D25 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D25. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D25. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D25. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-525. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D26

Offset	0x0418 (IOMUXC_SW_PAD_CTL_PAD_EIM_D26)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-526. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D26 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D26. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D26. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D26. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D26. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-526. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D26 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D26. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D26. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D26. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-527. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D27

Offset	0x041c (IOMUXC_SW_PAD_CTL_PAD_EIM_D27)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-528. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D27 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D27. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D27. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D27. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D27. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-528. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D27 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_D27. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D27. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D27. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-529. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D28

Offset	0x0420 (IOMUXC_SW_PAD_CTL_PAD_EIM_D28)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-530. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D28 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D28. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D28. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D28. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D28. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-530. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D28 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D28. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D28. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-531. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D29

Offset	0x0424 (IOMUXC_SW_PAD_CTL_PAD_EIM_D29)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-532. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D29 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D29. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D29. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D29. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D29. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-532. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D29 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D29. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D29. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-533. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D30

Offset	0x0428 (IOMUXC_SW_PAD_CTL_PAD_EIM_D30)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-534. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D30 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D30. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D30. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D30. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D30. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-534. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D30 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D30. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D30. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-535. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_D31

Offset	0x042c (IOMUXC_SW_PAD_CTL_PAD_EIM_D31)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-536. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D31 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_D31. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_D31. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_D31. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_D31. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-536. Register IOMUXC_SW_PAD_CTL_PAD_EIM_D31 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_D31. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_D31. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-537. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A16

Offset	0x0430 (IOMUXC_SW_PAD_CTL_PAD_EIM_A16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-538. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A16 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A16. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A16. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A16. 0: Keeper 1: Pull
5-0	Reserved

Table A-539. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A17

Offset	0x0434 (IOMUXC_SW_PAD_CTL_PAD_EIM_A17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-540. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A17 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A17. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A17. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A17. 0: Keeper 1: Pull
5-0	Reserved

Table A-541. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A18

Offset	0x0438 (IOMUXC_SW_PAD_CTL_PAD_EIM_A18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-542. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A18 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A18. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A18. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A18. 0: Keeper 1: Pull
5-0	Reserved

Table A-543. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A19

Offset: 0x043c (IOMUXC_SW_PAD_CTL_PAD_EIM_A19) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-544. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A19 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A19. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A19. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A19. 0: Keeper 1: Pull
5-0	Reserved

Table A-545. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A20

Offset	0x0440 (IOMUXC_SW_PAD_CTL_PAD_EIM_A20)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-546. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A20 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A20. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A20. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A20. 0: Keeper 1: Pull
5-0	Reserved

Table A-547. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A21

Offset	0x0444 (IOMUXC_SW_PAD_CTL_PAD_EIM_A21)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-548. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A21 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A21. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A21. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A21. 0: Keeper 1: Pull
5-0	Reserved

Table A-549. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A22

Offset	0x0448 (IOMUXC_SW_PAD_CTL_PAD_EIM_A22)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-550. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A22 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A22. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A22. 0: Keeper 1: Pull
5-0	Reserved

Table A-551. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A23

Offset	0x044c (IOMUXC_SW_PAD_CTL_PAD_EIM_A23)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-552. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A23 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A23. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A23. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A23. 0: Keeper 1: Pull
5-0	Reserved

Table A-553. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A24

Offset	0x0450 (IOMUXC_SW_PAD_CTL_PAD_EIM_A24)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-554. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A24 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A24. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A24. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A24. 0: Keeper 1: Pull
5-0	Reserved

Table A-555. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A25

Offset	0x0454 (IOMUXC_SW_PAD_CTL_PAD_EIM_A25)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-556. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A25 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A25. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A25. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A25. 0: Keeper 1: Pull
5-0	Reserved

Table A-557. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A26

Offset	0x0458 (IOMUXC_SW_PAD_CTL_PAD_EIM_A26)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Table A-558. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A26 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A26. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A26. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_A26. 0: Keeper 1: Pull
5-0	Reserved

Table A-559. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_A27

Offset	0x045c (IOMUXC_SW_PAD_CTL_PAD_EIM_A27)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-560. Register IOMUXC_SW_PAD_CTL_PAD_EIM_A27 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_A27. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_A27. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-561. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_EB0

Offset	0x0460 (IOMUXC_SW_PAD_CTL_PAD_EIM_EB0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-562. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_EB0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_EB0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_EB0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-563. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_EB1

Offset	0x0464 (IOMUXC_SW_PAD_CTL_PAD_EIM_EB1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-564. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_EB1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_EB1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_EB1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-565. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_EB2

Offset	0x0468 (IOMUXC_SW_PAD_CTL_PAD_EIM_EB2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-566. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_EB2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_EB2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_EB2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_EB2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-566. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB2 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: EIM_EB2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_EB2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_EB2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-567. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_EB3

Offset	0x046c (IOMUXC_SW_PAD_CTL_PAD_EIM_EB3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-568. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_EB3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_EB3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_EB3. 0: Keeper 1: Pull
5-3	Reserved

Table A-568. Register IOMUXC_SW_PAD_CTL_PAD_EIM_EB3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_EB3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_EB3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-569. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_OE

Offset: 0x0470 (IOMUXC_SW_PAD_CTL_PAD_EIM_OE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-570. Register IOMUXC_SW_PAD_CTL_PAD_EIM_OE Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_OE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_OE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_OE. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-571. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS0

Offset: 0x0474 (IOMUXC_SW_PAD_CTL_PAD_EIM_CS0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-572. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-573. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS1

Offset: 0x0478 (IOMUXC_SW_PAD_CTL_PAD_EIM_CS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-574. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-575. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS2

Offset	0x047c (IOMUXC_SW_PAD_CTL_PAD_EIM_CS2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-576. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_CS2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_CS2. 0: Keeper 1: Pull
5-3	Reserved

Table A-576. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-577. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS3

Offset: 0x0480 (IOMUXC_SW_PAD_CTL_PAD_EIM_CS3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-578. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_CS3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_CS3. 0: Keeper 1: Pull
5-3	Reserved

Table A-578. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-579. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS4

Offset: 0x0484 (IOMUXC_SW_PAD_CTL_PAD_EIM_CS4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-580. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_CS4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_CS4. 0: Keeper 1: Pull
5-3	Reserved

Table A-580. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS4 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-581. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CS5

Offset	0x0488 (IOMUXC_SW_PAD_CTL_PAD_EIM_CS5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-582. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS5 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_CS5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CS5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_CS5. 0: Keeper 1: Pull
5-3	Reserved

Table A-582. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CS5 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CS5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CS5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-583. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_DTACK

Offset: 0x048c (IOMUXC_SW_PAD_CTL_PAD_EIM_DTACK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-584. Register IOMUXC_SW_PAD_CTL_PAD_EIM_DTACK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_DTACK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_DTACK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_DTACK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-585. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_WAIT

Offset	0x0490 (IOMUXC_SW_PAD_CTL_PAD_EIM_WAIT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-586. Register IOMUXC_SW_PAD_CTL_PAD_EIM_WAIT Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_WAIT. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_WAIT. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-587. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_LBA

Offset: 0x0494 (IOMUXC_SW_PAD_CTL_PAD_EIM_LBA) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-588. Register IOMUXC_SW_PAD_CTL_PAD_EIM_LBA Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_LBA. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_LBA. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_LBA. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-589. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_BCLK

Offset: 0x0498 (IOMUXC_SW_PAD_CTL_PAD_EIM_BCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-590. Register IOMUXC_SW_PAD_CTL_PAD_EIM_BCLK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_BCLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_BCLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_BCLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-591. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_RW

Offset: 0x049c (IOMUXC_SW_PAD_CTL_PAD_EIM_RW) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-592. Register IOMUXC_SW_PAD_CTL_PAD_EIM_RW Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_RW. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_RW. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_RW. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-593. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_CRE

Offset	0x04a0 (IOMUXC_SW_PAD_CTL_PAD_EIM_CRE)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-594. Register IOMUXC_SW_PAD_CTL_PAD_EIM_CRE Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_CRE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_CRE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_CRE. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-595. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS

Offset: 0x04a4 (IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-596. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_RAS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_RAS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-597. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS

Offset: 0x04a8 (IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-598. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_CAS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_CAS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-599. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE

Offset: 0x04ac (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-600. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDWE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDWE. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDWE. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-600. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDWE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDWE. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-601. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0

Offset: 0x04b0 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-602. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDCKE0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDCKE0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDCKE0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-602. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDCKE0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDCKE0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-603. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1

Offset: 0x04b4 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-604. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDCKE1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDCKE1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDCKE1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-604. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDCKE1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDCKE1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-605. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK

Offset: 0x04b8 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-606. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDCLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDCLK. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDCLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-606. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDCLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDCLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-607. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0

Offset: 0x04bc (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-608. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DRAM_SDQS0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDQS0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDQS0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDQS0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-608. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDQS0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDQS0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-609. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1

Offset: 0x04c0 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-610. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DRAM_SDQS1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDQS1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDQS1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDQS1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-610. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDQS1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDQS1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-611. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2

Offset: 0x04c4 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-612. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DRAM_SDQS2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDQS2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDQS2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDQS2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-612. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDQS2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDQS2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-613. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3

Offset: 0x04c8 (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-614. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DRAM_SDQS3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_SDQS3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_SDQS3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_SDQS3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-614. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_SDQS3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_SDQS3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-615. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0

Offset: 0x04cc (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1[1]

[1] The value of this bit also controls DRAM_CS1slew rate.

Table A-616. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_CS0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_CS0. 0: Keeper 1: Pull

Table A-616. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_CS0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_CS0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_CS0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-617. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1

Offset: 0x04d0 (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0[1]

[1] The value of this bit is controlled by the output of the DRAM_CS0 SRE bit.

Table A-618. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_CS1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_CS1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_CS1. 00: 100 KOhm Pull Down 01: 47K Ohm Pull Up 10: 100K Ohm Pull Up 11: 22K Ohm Pull Up
3	Reserved

Table A-618. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_CS1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field. Select one out of next values for BGA contact: EIM_D16. 0: Slow Slew Rate 1: Fast Slew Rate **Note:** DRAM_CS0 slew-rate also controls the DRAM_CS1 slew-rate.

Table A-619. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0

Offset: 0x04d4 (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-620. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_DQM0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_DQM0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_DQM0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-620. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_DQM0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_DQM0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-621. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1

Offset: 0x04d8 (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-622. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_DQM1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_DQM1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_DQM1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-622. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_DQM1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_DQM1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-623. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2

Offset: 0x04dc (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-624. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_DQM2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_DQM2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_DQM2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-624. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_DQM2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_DQM2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-625. Register: IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3

Offset: 0x04e0 (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-626. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DRAM_DQM3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DRAM_DQM3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DRAM_DQM3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-626. Register IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DRAM_DQM3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DRAM_DQM3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-627. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_WE_B

Offset: 0x04e4 (IOMUXC_SW_PAD_CTL_PAD_NANDF_WE_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	0	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-628. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_WE_B Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_WE_B. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field. Select one out of next values for BGA contact: NANDF_WE_B. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_WE_B. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_WE_B. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-628. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_WE_B Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_WE_B. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-629. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RE_B

Offset: 0x04e8 (IOMUXC_SW_PAD_CTL_PAD_NANDF_RE_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	0	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-630. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RE_B Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_RE_B. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RE_B. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RE_B. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RE_B. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-630. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RE_B Bits Description

Field	Description
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RE_B. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-631. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_ALE

Offset	0x04ec (IOMUXC_SW_PAD_CTL_PAD_NANDF_ALE)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	0	0	0	0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-632. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_ALE Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_ALE. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_ALE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_ALE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-633. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CLE

Offset: 0x04f0 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CLE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	0	0	0	0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-634. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CLE Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CLE. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CLE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CLE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-635. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_WP_B

Offset: 0x04f4 (IOMUXC_SW_PAD_CTL_PAD_NANDF_WP_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	0	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-636. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_WP_B Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_WP_B. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_WP_B. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_WP_B. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_WP_B. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-636. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_WP_B Bits Description

Field	Description
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_WP_B. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-637. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RB0

Offset: 0x04f8 (IOMUXC_SW_PAD_CTL_PAD_NANDF_RB0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-638. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB0 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_RB0. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RB0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RB0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_RB0. 0: Keeper 1: Pull

Table A-638. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB0 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RB0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RB0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-639. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RB1

Offset: 0x04fc (IOMUXC_SW_PAD_CTL_PAD_NANDF_RB1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-640. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB1 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_RB1. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RB1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RB1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_RB1. 0: Keeper 1: Pull

Table A-640. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB1 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RB1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_RB1. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RB1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-641. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RB2

Offset: 0x0500 (IOMUXC_SW_PAD_CTL_PAD_NANDF_RB2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-642. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB2 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for: NANDF_RB2. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RB2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RB2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_RB2. 0: Keeper 1: Pull

Table A-642. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB2 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RB2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_RB2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RB2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-643. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RB3

Offset: 0x0504 (IOMUXC_SW_PAD_CTL_PAD_NANDF_RB3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-644. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB3 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_RB3. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RB3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RB3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_RB3. 0: Keeper 1: Pull

Table A-644. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RB3 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RB3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_RB3. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RB3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-645. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_SDBA2

Offset: 0x0508 (IOMUXC_SW_PAD_CTL_PAD_EIM_SDBA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	Reserved	0	0	0	DDR_INPUT	HYS	PKE	PUE	PUS		0	0	0	0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-646. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDBA2 Bits Description

Field	Description
31-14	Reserved
13	Reserved
12-10	Reserved
9 DDR_INPUT	DDR / CMOS Input Mode Field Select one out of next values for BGA contact: EIM_SDBA2. 0: CMOS input type 1: DDR2 input type
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_SDBA2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_SDBA2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled

Table A-646. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDBA2 Bits Description

Field	Description
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_SDBA2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_SDBA2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-647. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT1

Offset: 0x050c (IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0		0	0	0							0			
W			Reserved				DDR_INPUT	HYS	PKE	PUE	PUS			DSE		SRE
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	1

Table A-648. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT1 Bits Description

Field	Description
31-14	Reserved
13	Reserved
12-10	Reserved
9 DDR_INPUT	DDR / CMOS Input Mode Field Select one out of next values for BGA contact: EIM_SDODT1. 0: CMOS input type 1: DDR2 input type
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_SDODT1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_SDODT1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled

Table A-648. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT1 Bits Description

Field	Description
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_SDODT1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_SDODT1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_SDODT1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_SDODT1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-649. Register: IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT0

Offset: 0x0510 (IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	Reserved	0	0	0	DDR_INPUT	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	1

Table A-650. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT0 Bits Description

Field	Description
31-14	Reserved
13	Reserved
12-10	Reserved
9 DDR_INPUT	DDR / CMOS Input Mode Field Select one out of next values for BGA contact: EIM_SDODT0. 0: CMOS input type 1: DDR2 input type
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: EIM_SDODT0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: EIM_SDODT0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled

Table A-650. Register IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT0 Bits Description

Field	Description
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: EIM_SDODT0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: EIM_SDODT0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: EIM_SDODT0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: EIM_SDODT0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-651. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO_NAND

Offset: 0x0514 (IOMUXC_SW_PAD_CTL_PAD_GPIO_NAND) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	0	0	0	0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-652. Register IOMUXC_SW_PAD_CTL_PAD_GPIO_NAND Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for GPIO_NAND. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO_NAND. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO_NAND. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-653. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS0

Offset 0x0518 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS0) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-654. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-655. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS1

Offset: 0x051c (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-656. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS1 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-657. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS2

Offset: 0x0520 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-658. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS2 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CS2. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_CS2. 0: Keeper 1: Pull

Table A-658. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS2 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_CS2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-659. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS3

Offset: 0x0524 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-660. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS3 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CS3. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_CS3. 0: Keeper 1: Pull

Table A-660. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS3 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_CS3. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-661. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS4

Offset: 0x0528 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-662. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS4 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CS4. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_CS4. 0: Keeper 1: Pull

Table A-662. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS4 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS4. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-663. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS5

Offset: 0x052c (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-664. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS5 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CS5. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_CS5. 0: Keeper 1: Pull

Table A-664. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS5 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS5. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-665. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS6

Offset: 0x0530 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	0	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0

Table A-666. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS6 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_CS6. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS6. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-666. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS6 Bits Description

Field	Description
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-667. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_CS7

Offset: 0x0534 (IOMUXC_SW_PAD_CTL_PAD_NANDF_CS7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	0	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-668. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS7 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode forNANDF_CS7. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_CS7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_CS7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_CS7. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-668. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_CS7 Bits Description

Field	Description
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_CS7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-669. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_RDY_INT

Offset 0x0538 (IOMUXC_SW_PAD_CTL_PAD_NANDF_RDY_INT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-670. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RDY_INT Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_RDY_INT. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_RDY_INT. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_RDY_INT. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_RDY_INT. 0: Keeper 1: Pull

Table A-670. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_RDY_INT Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_RDY_INT. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_RDY_INT. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_RDY_INT. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-671. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D15

Offset: 0x053c (IOMUXC_SW_PAD_CTL_PAD_NANDF_D15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-672. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D15 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D15. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D15. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D15. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D15. 0: Keeper 1: Pull

Table A-672. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D15 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D15. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D15. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-673. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D14

Offset	0x0540 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-674. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D14 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D14. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D14. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D14. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D14. 0: Keeper 1: Pull

Table A-674. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D14 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D14. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D14. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-675. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D13

Offset	0x0544 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-676. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D13 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D13. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D13. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D13. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D13. 0: Keeper 1: Pull

Table A-676. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D13 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D13. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D13. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-677. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D12

Offset: 0x0548 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-678. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D12 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D12. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D12. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D12. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D12. 0: Keeper 1: Pull

Table A-678. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D12 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D12. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D12. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-679. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D11

Offset: 0x054c (IOMUXC_SW_PAD_CTL_PAD_NANDF_D11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-680. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D11 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D11. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D11. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D11. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D11. 0: Keeper 1: Pull

Table A-680. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D11 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D11. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D11. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-681. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D10

Offset	0x0550 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D10)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-682. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D10 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D10. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D10. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D10. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D10. 0: Keeper 1: Pull

Table A-682. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D10 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D10. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D10. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-683. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D9

Offset: 0x0554 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-684. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D9 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D9. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D9. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D9. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D9. 0: Keeper 1: Pull

Table A-684. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D9 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D9. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D9. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-685. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D8

Offset: 0x0558 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-686. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D8 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D8. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D8. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D8. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D8. 0: Keeper 1: Pull

Table A-686. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D8 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D8. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D8. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-687. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D7

Offset: 0x055c (IOMUXC_SW_PAD_CTL_PAD_NANDF_D7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-688. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D7 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D7. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D7. 0: Keeper 1: Pull

Table A-688. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D7 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D7. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-689. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D6

Offset: 0x0560 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-690. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D6 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D6. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D6. 0: Keeper 1: Pull

Table A-690. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D6 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D6. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: NANDF_D6. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-691. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D5

Offset: 0x0564 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-692. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D5 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D5. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D5. 0: Keeper 1: Pull

Table A-692. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D5 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D5. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-693. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D4

Offset: 0x0568 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-694. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D4 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D4. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D4. 0: Keeper 1: Pull

Table A-694. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D4 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D4. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-695. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D3

Offset: 0x056c (IOMUXC_SW_PAD_CTL_PAD_NANDF_D3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-696. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D3 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D3. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D3. 0: Keeper 1: Pull

Table A-696. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D3 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-697. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D2

Offset: 0x0570 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-698. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D2 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D2. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D2. 0: Keeper 1: Pull

Table A-698. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D2 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-699. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D1

Offset: 0x0574 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-700. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D1 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D1. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D1. 0: Keeper 1: Pull

Table A-700. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D1 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-701. Register: IOMUXC_SW_PAD_CTL_PAD_NANDF_D0

Offset: 0x0578 (IOMUXC_SW_PAD_CTL_PAD_NANDF_D0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	0	1	0	0	1	0	0

Table A-702. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D0 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for NANDF_D0. 0 High output voltage mode 1 Low output voltage mode **Note:** This bit also serves as “HVE” setting for pads NANDF_CS0 and NANDF_CS1, which have no dedicated bits for that purpose in their corresponding IOMUXC_SW_PAD_CTL registers
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: NANDF_D0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: NANDF_D0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: NANDF_D0. 0: Keeper 1: Pull

Table A-702. Register IOMUXC_SW_PAD_CTL_PAD_NANDF_D0 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: NANDF_D0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: NANDF_D0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-703. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D8

Offset: 0x057c (IOMUXC_SW_PAD_CTL_PAD_CSI1_D8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-704. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D8 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D8. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI1_D8. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI1_D8. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI1_D8. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-705. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D9

Offset	0x0580 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D9)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-706. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D9 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D9. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI1_D9. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI1_D9. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI1_D9. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-707. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D10

Offset	0x0584 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D10)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-708. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D10 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D10. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-709. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D11

Offset	0x0588 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D11)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-710. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D11 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D11. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-711. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D12

Offset	0x058c (IOMUXC_SW_PAD_CTL_PAD_CSI1_D12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-712. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D12 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D12. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-713. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D13

Offset	0x0590 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
W								HYS								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-714. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D13 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D13. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-715. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D14

Offset	0x0594 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-716. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D14 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D14. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-717. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D15

Offset	0x0598 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-718. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D15 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D15. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-719. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D16

Offset	0x059c (IOMUXC_SW_PAD_CTL_PAD_CSI1_D16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-720. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D16 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D16. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-721. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D17

Offset	0x05a0 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-722. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D17 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D17. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-723. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D18

Offset	0x05a4 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-724. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D18 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D18. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-725. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_D19

Offset	0x05a8 (IOMUXC_SW_PAD_CTL_PAD_CSI1_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-726. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_D19 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_D19. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-727. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_VSYNC

Offset: 0x05ac (IOMUXC_SW_PAD_CTL_PAD_CSI1_VSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-728. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_VSYNC Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_VSYNC. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI1_VSYNC. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-729. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_HSYNC

Offset: 0x05b0 (IOMUXC_SW_PAD_CTL_PAD_CSI1_HSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-730. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_HSYNC Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_HSYNC. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI1_HSYNC. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-731. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_PIXCLK

Offset	0x05b4 (IOMUXC_SW_PAD_CTL_PAD_CSI1_PIXCLK)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-732. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_PIXCLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI1_PIXCLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-733. Register: IOMUXC_SW_PAD_CTL_PAD_CSI1_MCLK

Offset: 0x05b8 (IOMUXC_SW_PAD_CTL_PAD_CSI1_MCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-734. Register IOMUXC_SW_PAD_CTL_PAD_CSI1_MCLK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI1_MCLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI1_MCLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI1_MCLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-735. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D12

Offset	0x05bc (IOMUXC_SW_PAD_CTL_PAD_CSI2_D12)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-736. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D12 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D12. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_D12. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_D12. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_D12. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-737. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D13

Offset	0x05c0 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-738. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D13 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D13. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_D13. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_D13. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_D13. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-739. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D14

Offset	0x05c4 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-740. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D14 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D14. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-741. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D15

Offset	0x05c8 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-742. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D15 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D15. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-743. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D16

Offset	0x05cc (IOMUXC_SW_PAD_CTL_PAD_CSI2_D16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-744. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D16 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D16. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-745. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D17

Offset	0x05d0 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D17)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-746. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D17 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D17. 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-747. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D18

Offset	0x05d4 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-748. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D18 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D18. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_D18. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_D18. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_D18. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-749. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_D19

Offset	0x05d8 (IOMUXC_SW_PAD_CTL_PAD_CSI2_D19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-750. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_D19 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_D19. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_D19. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_D19. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_D19. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-751. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_VSYNC

Offset: 0x05dc (IOMUXC_SW_PAD_CTL_PAD_CSI2_VSYNC) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-752. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_VSYNC Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_VSYNC. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_VSYNC. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_VSYNC. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_VSYNC. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-753. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_HSYNC

Offset	0x05e0 (IOMUXC_SW_PAD_CTL_PAD_CSI2_HSYNC)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Table A-754. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_HSYNC Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_HSYNC. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_HSYNC. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_HSYNC. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_HSYNC. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-755. Register: IOMUXC_SW_PAD_CTL_PAD_CSI2_PIXCLK

Offset: 0x05e4 (IOMUXC_SW_PAD_CTL_PAD_CSI2_PIXCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1

Table A-756. Register IOMUXC_SW_PAD_CTL_PAD_CSI2_PIXCLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSI2_PIXCLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSI2_PIXCLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSI2_PIXCLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSI2_PIXCLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-757. Register: IOMUXC_SW_PAD_CTL_PAD_I2C1_CLK

Offset	0x05e8 (IOMUXC_SW_PAD_CTL_PAD_I2C1_CLK)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	PUS		ODE	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-758. Register IOMUXC_SW_PAD_CTL_PAD_I2C1_CLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: I2C1_CLK. 0: Hysteresis Disabled 1: Hysteresis Enabled Note: In HSI2C pads Hyst. Enable control is tied to NVCC_I2C supply, so that changing its configuration in iomuxc register will have no effect.
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: I2C1_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: I2C1_CLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-758. Register IOMUXC_SW_PAD_CTL_PAD_I2C1_CLK Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: I2C1_CLK. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: I2C1_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-759. Register: IOMUXC_SW_PAD_CTL_PAD_I2C1_DAT

Offset	0x05ec (IOMUXC_SW_PAD_CTL_PAD_I2C1_DAT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	PUS		ODE	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0

Table A-760. Register IOMUXC_SW_PAD_CTL_PAD_I2C1_DAT Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: I2C1_DAT. 0: Hysteresis Disabled 1: Hysteresis Enabled Note: In HSI2C pads Hyst. Enable control is tied to NVCC_I2C supply, so that changing its configuration in iomuxc register will have no effect.
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: I2C1_DAT. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: I2C1_DAT. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-760. Register IOMUXC_SW_PAD_CTL_PAD_I2C1_DAT Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: I2C1_DAT. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: I2C1_DAT. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-761. Register: IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_TXD

Offset: 0x05f0 (IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_TXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-762. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_TXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: AUD3_BB_TXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: AUD3_BB_TXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: AUD3_BB_TXD. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: AUD3_BB_TXD. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-762. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_TXD Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: AUD3_BB_TXD. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: AUD3_BB_TXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: AUD3_BB_TXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-763. Register: IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_RXD

Offset: 0x05f4 (IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_RXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-764. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_RXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: AUD3_BB_RXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: AUD3_BB_RXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: AUD3_BB_RXD. 0: Keeper 1: Pull
5-3	Reserved

Table A-764. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_RXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: AUD3_BB_RXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: AUD3_BB_RXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-765. Register: IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_CK

Offset: 0x05f8 (IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_CK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-766. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_CK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: AUD3_BB_CK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: AUD3_BB_CK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: AUD3_BB_CK. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: AUD3_BB_CK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-766. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_CK Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: AUD3_BB_CK. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: AUD3_BB_CK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: AUD3_BB_CK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-767. Register: IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_FS

Offset: 0x05fc (IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_FS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-768. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_FS Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: AUD3_BB_FS. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: AUD3_BB_FS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: AUD3_BB_FS. 0: Keeper 1: Pull
5-3	Reserved

Table A-768. Register IOMUXC_SW_PAD_CTL_PAD_AUD3_BB_FS Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: AUD3_BB_FS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: AUD3_BB_FS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-769. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_MOSI

Offset	0x0600 (IOMUXC_SW_PAD_CTL_PAD_CSPI1_MOSI)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-770. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_MOSI Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_MOSI. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_MOSI. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: CSPI1_MOSI. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: CSPI1_MOSI. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-770. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_MOSI Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_MOSI. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_MOSI. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-771. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_MISO

Offset	0x0604 (IOMUXC_SW_PAD_CTL_PAD_CSPI1_MISO)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-772. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_MISO Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_MISO. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_MISO. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_MISO. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_MISO. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-773. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS0

Offset	0x0608 (IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-774. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_SS0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_SS0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_SS0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_SS0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-775. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS1

Offset: 0x060c (IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-776. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_SS1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_SS1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_SS1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_SS1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_SS1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-777. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_RDY

Offset	0x0610 (IOMUXC_SW_PAD_CTL_PAD_CSPI1_RDY)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-778. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_RDY Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_RDY. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_RDY. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: CSPI1_RDY. 0: Keeper 1: Pull
5-3	Reserved

Table A-778. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_RDY Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_RDY. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_RDY. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-779. Register: IOMUXC_SW_PAD_CTL_PAD_CSPI1_SCLK

Offset: 0x0614 (IOMUXC_SW_PAD_CTL_PAD_CSPI1_SCLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-780. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_SCLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: CSPI1_SCLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: CSPI1_SCLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: CSPI1_SCLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: CSPI1_SCLK. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-780. Register IOMUXC_SW_PAD_CTL_PAD_CSPI1_SCLK Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: CSPI1_SCLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: CSPI1_SCLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-781. Register: IOMUXC_SW_PAD_CTL_PAD_UART1_RXD

Offset: 0x0618 (IOMUXC_SW_PAD_CTL_PAD_UART1_RXD)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-782. Register IOMUXC_SW_PAD_CTL_PAD_UART1_RXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART1_RXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART1_RXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART1_RXD. 0: Keeper 1: Pull
5-3	Reserved

Table A-782. Register IOMUXC_SW_PAD_CTL_PAD_UART1_RXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART1_RXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART1_RXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-783. Register: IOMUXC_SW_PAD_CTL_PAD_UART1_TXD

Offset: 0x061c (IOMUXC_SW_PAD_CTL_PAD_UART1_TXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-784. Register IOMUXC_SW_PAD_CTL_PAD_UART1_TXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART1_TXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART1_TXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART1_TXD. 0: Keeper 1: Pull
5-3	Reserved

Table A-784. Register IOMUXC_SW_PAD_CTL_PAD_UART1_TXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART1_TXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART1_TXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-785. Register: IOMUXC_SW_PAD_CTL_PAD_UART1_RTS

Offset	0x0620 (IOMUXC_SW_PAD_CTL_PAD_UART1_RTS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-786. Register IOMUXC_SW_PAD_CTL_PAD_UART1_RTS Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART1_RTS. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART1_RTS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART1_RTS. 0: Keeper 1: Pull
5-3	Reserved

Table A-786. Register IOMUXC_SW_PAD_CTL_PAD_UART1_RTS Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART1_RTS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART1_RTS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-787. Register: IOMUXC_SW_PAD_CTL_PAD_UART1_CTS

Offset	0x0624 (IOMUXC_SW_PAD_CTL_PAD_UART1_CTS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-788. Register IOMUXC_SW_PAD_CTL_PAD_UART1_CTS Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART1_CTS. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART1_CTS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART1_CTS. 0: Keeper 1: Pull
5-3	Reserved

Table A-788. Register IOMUXC_SW_PAD_CTL_PAD_UART1_CTS Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART1_CTS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART1_CTS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-789. Register: IOMUXC_SW_PAD_CTL_PAD_UART2_RXD

Offset: 0x0628 (IOMUXC_SW_PAD_CTL_PAD_UART2_RXD)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-790. Register IOMUXC_SW_PAD_CTL_PAD_UART2_RXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART2_RXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART2_RXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART2_RXD. 0: Keeper 1: Pull
5-4	Reserved
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: UART2_RXD. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-790. Register IOMUXC_SW_PAD_CTL_PAD_UART2_RXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART2_RXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART2_RXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-791. Register: IOMUXC_SW_PAD_CTL_PAD_UART2_TXD

Offset: 0x062c (IOMUXC_SW_PAD_CTL_PAD_UART2_TXD) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-792. Register IOMUXC_SW_PAD_CTL_PAD_UART2_TXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART2_TXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART2_TXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART2_TXD. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: UART2_TXD. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-792. Register IOMUXC_SW_PAD_CTL_PAD_UART2_TXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART2_TXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART2_TXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-793. Register: IOMUXC_SW_PAD_CTL_PAD_UART3_RXD

Offset: 0x0630 (IOMUXC_SW_PAD_CTL_PAD_UART3_RXD)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-794. Register IOMUXC_SW_PAD_CTL_PAD_UART3_RXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART3_RXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART3_RXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART3_RXD. 0: Keeper 1: Pull
5-3	Reserved

Table A-794. Register IOMUXC_SW_PAD_CTL_PAD_UART3_RXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART3_RXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART3_RXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-795. Register: IOMUXC_SW_PAD_CTL_PAD_UART3_TXD

Offset: 0x0634 (IOMUXC_SW_PAD_CTL_PAD_UART3_TXD)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-796. Register IOMUXC_SW_PAD_CTL_PAD_UART3_TXD Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: UART3_TXD. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: UART3_TXD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: UART3_TXD. 0: Keeper 1: Pull
5-3	Reserved

Table A-796. Register IOMUXC_SW_PAD_CTL_PAD_UART3_TXD Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: UART3_TXD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: UART3_TXD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-797. Register: IOMUXC_SW_PAD_CTL_PAD_OWIRE_LINE

Offset: 0x0638 (IOMUXC_SW_PAD_CTL_PAD_OWIRE_LINE) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	0	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-798. Register IOMUXC_SW_PAD_CTL_PAD_OWIRE_LINE Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: OWIRE_LINE. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: OWIRE_LINE. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: OWIRE_LINE. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: OWIRE_LINE. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-798. Register IOMUXC_SW_PAD_CTL_PAD_OWIRE_LINE Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: OWIRE_LINE. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: OWIRE_LINE. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-799. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0

Offset	0x063c (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-800. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_ROW0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_ROW0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_ROW0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_ROW0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-800. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_ROW0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_ROW0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-801. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1

Offset: 0x0640 (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-802. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_ROW1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_ROW1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_ROW1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_ROW1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-802. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_ROW1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_ROW1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-803. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2

Offset: 0x0644 (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-804. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_ROW2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_ROW2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_ROW2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_ROW2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-804. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_ROW2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_ROW2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-805. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3

Offset	0x0648 (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-806. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_ROW3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_ROW3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_ROW3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_ROW3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-806. Register IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_ROW3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_ROW3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-807. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL0

Offset	0x064c (IOMUXC_SW_PAD_CTL_PAD_KEY_COL0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-808. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-808. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL0 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL0. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-809. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL1

Offset: 0x0650 (IOMUXC_SW_PAD_CTL_PAD_KEY_COL1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-810. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-810. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL1 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL1. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-811. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL2

Offset	0x0654 (IOMUXC_SW_PAD_CTL_PAD_KEY_COL2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-812. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-812. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL2 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-813. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL3

Offset	0x0658 (IOMUXC_SW_PAD_CTL_PAD_KEY_COL3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-814. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-814. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL3 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL3. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-815. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL4

Offset: 0x065c (IOMUXC_SW_PAD_CTL_PAD_KEY_COL4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-816. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL4. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL4. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-816. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL4 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL4. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-817. Register: IOMUXC_SW_PAD_CTL_PAD_KEY_COL5

Offset	0x0660 (IOMUXC_SW_PAD_CTL_PAD_KEY_COL5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-818. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL5 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: KEY_COL5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: KEY_COL5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: KEY_COL5. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: KEY_COL5. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-818. Register IOMUXC_SW_PAD_CTL_PAD_KEY_COL5 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: KEY_COL5. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: KEY_COL5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: KEY_COL5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-819. Register: IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK

Offset: 0x0664 (IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-820. Register IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: JTAG_TCK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-821. Register: IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS

Offset	0x0668 (IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-822. Register IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: JTAG_TMS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-823. Register: IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI

Offset	0x066c (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-824. Register IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: JTAG_TDI. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-825. Register: IOMUXC_SW_PAD_CTL_PAD_JTAG_TRSTB

Offset	0x0670 (IOMUXC_SW_PAD_CTL_PAD_JTAG_TRSTB)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-826. Register IOMUXC_SW_PAD_CTL_PAD_JTAG_TRSTB Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: JTAG_TRSTB. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-827. Register: IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD

Offset	0x0674 (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-828. Register IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for BGA contact: JTAG_MOD. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-829. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_CLK

Offset	0x0678 (IOMUXC_SW_PAD_CTL_PAD_USBH1_CLK)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-830. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_CLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_CLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_CLK. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: USBH1_CLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-830. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_CLK Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: USBH1_CLK. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_CLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-831. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DIR

Offset: 0x067c (IOMUXC_SW_PAD_CTL_PAD_USBH1_DIR) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-832. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DIR Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DIR. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DIR. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DIR. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: USBH1_DIR. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-832. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DIR Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: USBH1_DIR. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DIR. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DIR. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-833. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_STP

Offset: 0x0680 (IOMUXC_SW_PAD_CTL_PAD_USBH1_STP) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-834. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_STP Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_STP. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_STP. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_STP. 0: Keeper 1: Pull
5-3	Reserved

Table A-834. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_STP Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_STP. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_STP. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-835. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_NXT

Offset: 0x0684 (IOMUXC_SW_PAD_CTL_PAD_USBH1_NXT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-836. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_NXT Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_NXT. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_NXT. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_NXT. 0: Keeper 1: Pull
5-3	Reserved

Table A-836. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_NXT Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_NXT. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_NXT. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-837. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA0

Offset: 0x0688 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-838. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA0. 0: Keeper 1: Pull
5-3	Reserved

Table A-838. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-839. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA1

Offset	0x068c (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-840. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA1. 0: Keeper 1: Pull
5-3	Reserved

Table A-840. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-841. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA2

Offset	0x0690 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-842. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA2. 0: Keeper 1: Pull
5-3	Reserved

Table A-842. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-843. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA3

Offset: 0x0694 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-844. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA3. 0: Keeper 1: Pull
5-3	Reserved

Table A-844. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-845. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA4

Offset	0x0698 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-846. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA4. 0: Keeper 1: Pull
5-3	Reserved

Table A-846. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA4 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-847. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA5

Offset: 0x069c (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-848. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA5 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA5. 0: Keeper 1: Pull
5-3	Reserved

Table A-848. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA5 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-849. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA6

Offset: 0x06a0 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-850. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA6 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA6. 0: Keeper 1: Pull
5-3	Reserved

Table A-850. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA6 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA6. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-851. Register: IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA7

Offset: 0x06a4 (IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-852. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA7 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: USBH1_DATA7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: USBH1_DATA7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: USBH1_DATA7. 0: Keeper 1: Pull
5-3	Reserved

Table A-852. Register IOMUXC_SW_PAD_CTL_PAD_USBH1_DATA7 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: USBH1_DATA7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: USBH1_DATA7. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-853. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN11

Offset	0x06a8 (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN11)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-854. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN11 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI1_PIN11. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN11. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_PIN11. 0: Keeper 1: Pull
5-3	Reserved

Table A-854. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN11 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN11. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN11. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-855. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN12

Offset: 0x06ac (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-856. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN12 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN12. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_PIN12. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN12. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN12. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-857. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN13

Offset	0x06b0 (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN13)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-858. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN13 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN13. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_PIN13. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN13. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN13. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-859. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_D0_CS

Offset: 0x06b4 (IOMUXC_SW_PAD_CTL_PAD_DI1_D0_CS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0			0	0	0			
W									PKE	PUE				DSE		SRE
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-860. Register IOMUXC_SW_PAD_CTL_PAD_DI1_D0_CS Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_D0_CS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_D0_CS. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_D0_CS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_D0_CS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-861. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_D1_CS

Offset	0x06b8 (IOMUXC_SW_PAD_CTL_PAD_DI1_D1_CS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-862. Register IOMUXC_SW_PAD_CTL_PAD_DI1_D1_CS Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_D1_CS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_D1_CS. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_D1_CS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_D1_CS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-863. Register: IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIN

Offset: 0x06bc (IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIN) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-864. Register IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIN Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISPB2_SER_DIN. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISPB2_SER_DIN. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISPB2_SER_DIN. 0: Keeper 1: Pull
5-3	Reserved

Table A-864. Register IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIN Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISPB2_SER_DIN. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISPB2_SER_DIN. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-865. Register: IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIO

Offset: 0x06c0 (IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIO) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-866. Register IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_DIO Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISPB2_SER_DIO. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISPB2_SER_DIO. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISPB2_SER_DIO. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISPB2_SER_DIO. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-867. Register: IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_CLK

Offset: 0x06c4 (IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-868. Register IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_CLK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISPB2_SER_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISPB2_SER_CLK. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISPB2_SER_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISPB2_SER_CLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-869. Register: IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_RS

Offset: 0x06c8 (IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_RS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-870. Register IOMUXC_SW_PAD_CTL_PAD_DISPB2_SER_RS Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISPB2_SER_RS. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISPB2_SER_RS. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISPB2_SER_RS. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISPB2_SER_RS. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-871. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT0

Offset: 0x06cc (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-872. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT0 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT0. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-873. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT1

Offset: 0x06d0 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-874. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT1 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT1. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-875. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT2

Offset: 0x06d4 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-876. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT2 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT2. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-877. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT3

Offset: 0x06d8 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-878. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT3 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT3. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-879. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT4

Offset: 0x06dc (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-880. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT4 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT4. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-881. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT5

Offset 0x06e0 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT5) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-882. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT5 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT5. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-883. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT6

Offset: 0x06e4 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-884. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT6 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT6. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT6. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-884. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT6 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT6. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-885. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT7

Offset: 0x06e8 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-886. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT7 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT7. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT7. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-886. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT7 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT7. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-887. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT8

Offset: 0x06ec (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-888. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT8 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT8. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT8. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT8. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT8. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-888. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT8 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT8. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT8. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-889. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT9

Offset: 0x06f0 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-890. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT9 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT9. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT9. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT9. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT9. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-890. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT9 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT9. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT9. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-891. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT10

Offset: 0x06f4 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-892. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT10 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT10. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT10. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT10. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT10. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-892. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT10 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT10. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT10. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-893. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT11

Offset: 0x06f8 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-894. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT11 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT11. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT11. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT11. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT11. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-894. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT11 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT11. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT11. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-895. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT12

Offset: 0x06fc (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-896. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT12 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT12. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT12. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT12. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT12. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-896. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT12 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT12. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT12. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-897. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT13

Offset	0x0700 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT13)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-898. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT13 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT13. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT13. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT13. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT13. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-898. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT13 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT13. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT13. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-899. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT14

Offset	0x0704 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT14)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-900. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT14 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT14. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT14. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT14. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT14. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-900. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT14 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT14. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT14. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-901. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT15

Offset: 0x0708 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-902. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT15 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT15. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT15. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT15. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT15. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-902. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT15 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT15. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT15. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-903. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT16

Offset	0x070c (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT16)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-904. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT16 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT16. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT16. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT16. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT16. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-904. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT16 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT16. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT16. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-905. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT17

Offset: 0x0710 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT17) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-906. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT17 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT17. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT17. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT17. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT17. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-906. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT17 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT17. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT17. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-907. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT18

Offset	0x0714 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT18)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-908. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT18 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT18. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT18. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT18. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT18. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-908. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT18 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT18. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT18. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-909. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT19

Offset	0x0718 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT19)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-910. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT19 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT19. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT19. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT19. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT19. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-910. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT19 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT19. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT19. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-911. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT20

Offset: 0x071c (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT20) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-912. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT20 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT20. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT20. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT20. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT20. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-912. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT20 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT20. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT20. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-913. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT21

Offset	0x0720 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT21)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-914. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT21 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT21. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT21. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT21. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT21. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-914. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT21 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT21. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT21. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-915. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT22

Offset: 0x0724 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT22) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-916. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT22 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT22. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT22. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT22. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT22. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-916. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT22 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT22. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT22. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-917. Register: IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT23

Offset: 0x0728 (IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT23) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-918. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT23 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP1_DAT23. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP1_DAT23. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP1_DAT23. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: DISP1_DAT23. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-918. Register IOMUXC_SW_PAD_CTL_PAD_DISP1_DAT23 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP1_DAT23. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP1_DAT23. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-919. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN3

Offset	0x072c (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-920. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI1_PIN3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_PIN3. 0: Keeper 1: Pull
5-3	Reserved

Table A-920. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-921. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_DISP_CLK

Offset	0x0730 (IOMUXC_SW_PAD_CTL_PAD_DI1_DISP_CLK)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-922. Register IOMUXC_SW_PAD_CTL_PAD_DI1_DISP_CLK Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_DISP_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_DISP_CLK. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_DISP_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_DISP_CLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-923. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN2

Offset	0x0734 (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-924. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI1_PIN2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI1_PIN2. 0: Keeper 1: Pull
5-3	Reserved

Table A-924. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-925. Register: IOMUXC_SW_PAD_CTL_PAD_DI1_PIN15

Offset: 0x0738 (IOMUXC_SW_PAD_CTL_PAD_DI1_PIN15) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-926. Register IOMUXC_SW_PAD_CTL_PAD_DI1_PIN15 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI1_PIN15. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI1_PIN15. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI1_PIN15. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-927. Register: IOMUXC_SW_PAD_CTL_PAD_DI_GP1

Offset: 0x073c (IOMUXC_SW_PAD_CTL_PAD_DI_GP1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-928. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI_GP1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI_GP1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI_GP1. 0: Keeper 1: Pull
5-3	Reserved

Table A-928. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI_GP1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI_GP1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-929. Register: IOMUXC_SW_PAD_CTL_PAD_DI_GP2

Offset: 0x0740 (IOMUXC_SW_PAD_CTL_PAD_DI_GP2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-930. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI_GP2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI_GP2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI_GP2. 0: Keeper 1: Pull
5-3	Reserved

Table A-930. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI_GP2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI_GP2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-931. Register: IOMUXC_SW_PAD_CTL_PAD_DI_GP3

Offset: 0x0744 (IOMUXC_SW_PAD_CTL_PAD_DI_GP3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	1

Table A-932. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI_GP3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI_GP3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI_GP3. 0: Keeper 1: Pull
5-3	Reserved

Table A-932. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI_GP3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI_GP3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-933. Register: IOMUXC_SW_PAD_CTL_PAD_DI2_PIN4

Offset: 0x0748 (IOMUXC_SW_PAD_CTL_PAD_DI2_PIN4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-934. Register IOMUXC_SW_PAD_CTL_PAD_DI2_PIN4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI2_PIN4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI2_PIN4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI2_PIN4. 0: Keeper 1: Pull
5-3	Reserved

Table A-934. Register IOMUXC_SW_PAD_CTL_PAD_DI2_PIN4 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI2_PIN4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI2_PIN4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-935. Register: IOMUXC_SW_PAD_CTL_PAD_DI2_PIN2

Offset	0x074c (IOMUXC_SW_PAD_CTL_PAD_DI2_PIN2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-936. Register IOMUXC_SW_PAD_CTL_PAD_DI2_PIN2 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI2_PIN2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI2_PIN2. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI2_PIN2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI2_PIN2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-937. Register: IOMUXC_SW_PAD_CTL_PAD_DI2_PIN3

Offset	0x0750 (IOMUXC_SW_PAD_CTL_PAD_DI2_PIN3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-938. Register IOMUXC_SW_PAD_CTL_PAD_DI2_PIN3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI2_PIN3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI2_PIN3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI2_PIN3. 0: Keeper 1: Pull
5-4	Reserved
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: DI2_PIN3. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-938. Register IOMUXC_SW_PAD_CTL_PAD_DI2_PIN3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI2_PIN3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI2_PIN3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-939. Register: IOMUXC_SW_PAD_CTL_PAD_DI2_DISP_CLK

Offset: 0x0754 (IOMUXC_SW_PAD_CTL_PAD_DI2_DISP_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-940. Register IOMUXC_SW_PAD_CTL_PAD_DI2_DISP_CLK Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI2_DISP_CLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI2_DISP_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI2_DISP_CLK. 0: Keeper 1: Pull
5-3	Reserved

Table A-940. Register IOMUXC_SW_PAD_CTL_PAD_DI2_DISP_CLK Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI2_DISP_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI2_DISP_CLK. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-941. Register: IOMUXC_SW_PAD_CTL_PAD_DI_GP4

Offset: 0x0758 (IOMUXC_SW_PAD_CTL_PAD_DI_GP4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0

Table A-942. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DI_GP4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DI_GP4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DI_GP4. 0: Keeper 1: Pull
5-3	Reserved

Table A-942. Register IOMUXC_SW_PAD_CTL_PAD_DI_GP4 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DI_GP4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DI_GP4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-943. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT0

Offset	0x075c (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-944. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT0. 0: Keeper 1: Pull
5-4	Reserved
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: DISP2_DAT0. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-944. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-945. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT1

Offset: 0x0760 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-946. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT1. 0: Keeper 1: Pull
5-4	Reserved
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: DISP2_DAT1. 0: Open Drain Disabled 1: Open Drain Enabled

Table A-946. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-947. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT2

Offset: 0x0764 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-948. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT2 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT2. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-949. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT3

Offset: 0x0768 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-950. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT3 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT3. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-951. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT4

Offset: 0x076c (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-952. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT4 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT4. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-953. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT5

Offset: 0x0770 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Table A-954. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT5 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT5. 0: Keeper 1: Pull
5-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-955. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT6

Offset: 0x0774 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT6) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-956. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT6 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT6. 0: Keeper 1: Pull
5-3	Reserved

Table A-956. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT6 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT6. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-957. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT7

Offset: 0x0778 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT7) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-958. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT7 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT7. 0: Keeper 1: Pull
5-3	Reserved

Table A-958. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT7 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT7. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-959. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT8

Offset: 0x077c (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT8) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-960. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT8 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT8. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT8. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT8. 0: Keeper 1: Pull
5-3	Reserved

Table A-960. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT8 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT8. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT8. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-961. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT9

Offset: 0x0780 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT9) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-962. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT9 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT9. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT9. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT9. 0: Keeper 1: Pull
5-3	Reserved

Table A-962. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT9 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT9. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT9. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-963. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT10

Offset: 0x0784 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT10) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-964. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT10 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT10. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT10. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT10. 0: Keeper 1: Pull
5-3	Reserved

Table A-964. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT10 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT10. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT10. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-965. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT11

Offset: 0x0788 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT11) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-966. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT11 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT11. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT11. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT11. 0: Keeper 1: Pull
5-3	Reserved

Table A-966. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT11 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT11. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT11. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-967. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT12

Offset: 0x078c (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT12) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-968. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT12 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT12. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT12. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT12. 0: Keeper 1: Pull
5-3	Reserved

Table A-968. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT12 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT12. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT12. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-969. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT13

Offset: 0x0790 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT13) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-970. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT13 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT13. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT13. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT13. 0: Keeper 1: Pull
5-3	Reserved

Table A-970. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT13 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT13. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT13. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-971. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT14

Offset: 0x0794 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT14) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-972. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT14 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT14. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT14. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT14. 0: Keeper 1: Pull
5-3	Reserved

Table A-972. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT14 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT14. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT14. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-973. Register: IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT15

Offset	0x0798 (IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT15)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	0	0	0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-974. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT15 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: DISP2_DAT15. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: DISP2_DAT15. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: DISP2_DAT15. 0: Keeper 1: Pull
5-3	Reserved

Table A-974. Register IOMUXC_SW_PAD_CTL_PAD_DISP2_DAT15 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: DISP2_DAT15. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: DISP2_DAT15. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-975. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_CMD

Offset	0x079c (IOMUXC_SW_PAD_CTL_PAD_SD1_CMD)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	1	1	0	0

Table A-976. Register IOMUXC_SW_PAD_CTL_PAD_SD1_CMD Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD1_CMD. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_CMD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_CMD. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_CMD. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-976. Register IOMUXC_SW_PAD_CTL_PAD_SD1_CMD Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: SD1_CMD. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_CMD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-977. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_CLK

Offset: 0x07a0 (IOMUXC_SW_PAD_CTL_PAD_SD1_CLK) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-978. Register IOMUXC_SW_PAD_CTL_PAD_SD1_CLK Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode forSD1_CLK. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: SD1_CLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_CLK. 0: Keeper 1: Pull

Table A-978. Register IOMUXC_SW_PAD_CTL_PAD_SD1_CLK Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_CLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-979. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0

Offset: 0x07a4 (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-980. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD1_DATA0. 0 High output voltage mode 1 Low output voltage modee
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_DATA0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_DATA0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_DATA0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-980. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_DATA0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-981. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1

Offset	0x07a8 (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-982. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD1_DATA1. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_DATA1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_DATA1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_DATA1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-982. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_DATA1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-983. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2

Offset: 0x07ac (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-984. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD1_DATA2. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_DATA2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_DATA2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_DATA2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-984. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_DATA2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-985. Register: IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3

Offset	0x07b0 (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0

Table A-986. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD1_DATA3. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD1_DATA3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD1_DATA3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD1_DATA3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-986. Register IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD1_DATA3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-987. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_0

Offset	0x07b4 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	1

Table A-988. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_0. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-988. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_0. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-989. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_1

Offset	0x07b8 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	0	1	0	0	1	0	1

Table A-990. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_1. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-990. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_1 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_1. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-991. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_CMD

Offset	0x07bc (IOMUXC_SW_PAD_CTL_PAD_SD2_CMD)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	1	1	0	0

Table A-992. Register IOMUXC_SW_PAD_CTL_PAD_SD2_CMD Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD2_CMD. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_CMD. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_CMD. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_CMD. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-992. Register IOMUXC_SW_PAD_CTL_PAD_SD2_CMD Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: SD2_CMD. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_CMD. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-993. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_CLK

Offset	0x07c0 (IOMUXC_SW_PAD_CTL_PAD_SD2_CLK)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-994. Register IOMUXC_SW_PAD_CTL_PAD_SD2_CLK Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD2_CLK. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: SD2_CLK. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_CLK. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_CLK. 0: Keeper 1: Pull

Table A-994. Register IOMUXC_SW_PAD_CTL_PAD_SD2_CLK Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_CLK. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: SD2_CLK. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_CLK. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-995. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0

Offset: 0x07c4 (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	1	0	1	0	0

Table A-996. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD2_DATA0. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_DATA0. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_DATA0. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_DATA0. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-996. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_DATA0. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-997. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1

Offset: 0x07c8 (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	1	1	1	0	1	0	1	0	0

Table A-998. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode forSD2_DATA1. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: SD2_DATA1. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_DATA1. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_DATA1. 0: Keeper 1: Pull

Table A-998. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_DATA1. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: SD2_DATA1. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_DATA1. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-999. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2

Offset: 0x07cc (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		0
W																
Reset	0	0	1	0	0	0	0	1	1	1	0	1	0	1	0	0

Table A-1000. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD2_DATA2. 0 High output voltage mode 1 Low output voltage mode
12-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: SD2_DATA2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_DATA2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_DATA2. 0: Keeper 1: Pull

Table A-1000. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2 Bits Description

Field	Description
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_DATA2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: SD2_DATA2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_DATA2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1001. Register: IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3

Offset: 0x07d0 (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	PKE	PUE	PUS		0	DSE		0
W																
Reset	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0

Table A-1002. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3 Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for SD2_DATA3. 0 High output voltage mode 1 Low output voltage mode
12-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: SD2_DATA3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: SD2_DATA3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: SD2_DATA3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-1002. Register IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: SD2_DATA3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1003. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_2

Offset	0x07d4 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1004. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_2. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_2. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_2. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_2. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1004. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_2 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_2. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_2. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_2. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1005. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_3

Offset	0x07d8 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1006. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_3. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_3. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_3. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_3. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1006. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_3 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_3. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_3. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_3. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1007. Register: IOMUXC_SW_PAD_CTL_PAD_RESET_IN_B

Offset	0x07dc (IOMUXC_SW_PAD_CTL_PAD_RESET_IN_B)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1008. Register IOMUXC_SW_PAD_CTL_PAD_RESET_IN_B Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: RESET_IN_B. 0: Keeper 1: Pull
5-0	Reserved

Table A-1009. Register: IOMUXC_SW_PAD_CTL_PAD_POR_B

Offset: 0x07e0 (IOMUXC_SW_PAD_CTL_PAD_POR_B) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1010. Register IOMUXC_SW_PAD_CTL_PAD_POR_B Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: POR_B. 0: Keeper 1: Pull
5-0	Reserved

Table A-1011. Register: IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1

Offset	0x07e4 (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1012. Register IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE1 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: BOOT_MODE1. 0: Keeper 1: Pull
5-0	Reserved

Table A-1013. Register: IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0

Offset: 0x07e8 (IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1014. Register IOMUXC_SW_PAD_CTL_PAD_BOOT_MODE0 Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: BOOT_MODE0. 0: Keeper 1: Pull
5-0	Reserved

Table A-1015. Register: IOMUXC_SW_PAD_CTL_PAD_PMIC_RDY

Offset	0x07ec (IOMUXC_SW_PAD_CTL_PAD_PMIC_RDY)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	PUS		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Table A-1016. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_RDY Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: PMIC_RDY. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: PMIC_RDY. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-1017. Register: IOMUXC_SW_PAD_CTL_PAD_CKIL

Offset: 0x07f0 (IOMUXC_SW_PAD_CTL_PAD_CKIL) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	PUS		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Table A-1018. Register IOMUXC_SW_PAD_CTL_PAD_CKIL Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: CKIL. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: CKIL. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-1019. Register: IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ

Offset: 0x07f4 (IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	1

Table A-1020. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: PMIC_STBY_REQ. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: PMIC_STBY_REQ. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: PMIC_STBY_REQ. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-1020. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_STBY_REQ Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: PMIC_STBY_REQ. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: PMIC_STBY_REQ. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1021. Register: IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ

Offset	0x07f8 (IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	PUE	PUS		ODE	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0

Table A-1022. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_ON_REQ Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: PMIC_ON_REQ. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: PMIC_ON_REQ. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: PMIC_ON_REQ. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: PMIC_ON_REQ. 0: Open Drain Disabled 1: Open Drain Enabled
2-0	Reserved

Table A-1023. Register: IOMUXC_SW_PAD_CTL_PAD_PMIC_INT_REQ

Offset: 0x07fc (IOMUXC_SW_PAD_CTL_PAD_PMIC_INT_REQ) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	0	0	1	0	1

Table A-1024. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_INT_REQ Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: PMIC_INT_REQ. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: PMIC_INT_REQ. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: PMIC_INT_REQ. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: PMIC_INT_REQ. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-1024. Register IOMUXC_SW_PAD_CTL_PAD_PMIC_INT_REQ Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: PMIC_INT_REQ. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: PMIC_INT_REQ. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1025. Register: IOMUXC_SW_PAD_CTL_PAD_CLK_SS

Offset	0x0800 (IOMUXC_SW_PAD_CTL_PAD_CLK_SS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	PUS		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

Table A-1026. Register IOMUXC_SW_PAD_CTL_PAD_CLK_SS Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: CLK_SS. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: CLK_SS. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-1027. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_4

Offset	0x0804 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1028. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_4 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_4. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_4. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_4. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_4. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1028. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_4 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_4. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_4. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_4. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1029. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_5

Offset	0x0808 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_5)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1030. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_5 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_5. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_5. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_5. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_5. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1030. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_5 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_5. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_5. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_5. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1031. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_6

Offset	0x080c (IOMUXC_SW_PAD_CTL_PAD_GPIO1_6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1032. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_6 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_6. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_6. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_6. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_6. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1032. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_6 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_6. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_6. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_6. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1033. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_7

Offset	0x0810 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_7)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1034. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_7 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_7. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_7. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_7. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_7. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-1034. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_7 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_7. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_7. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1035. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_8

Offset	0x0814 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_8)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		0	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1

Table A-1036. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_8 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_8. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_8. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_8. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_8. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3	Reserved

Table A-1036. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_8 Bits Description

Field	Description
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_8. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_8. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1037. Register: IOMUXC_SW_PAD_CTL_PAD_GPIO1_9

Offset	0x0818 (IOMUXC_SW_PAD_CTL_PAD_GPIO1_9)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	PKE	PUE	PUS		ODE	DSE		SRE
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	1

Table A-1038. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_9 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for BGA contact: GPIO1_9. 0: Hysteresis Disabled 1: Hysteresis Enabled
7 PKE	Pull / Keep Enable Field Select one out of next values for BGA contact: GPIO1_9. 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6 PUE	Pull / Keep Select Field Select one out of next values for BGA contact: GPIO1_9. 0: Keeper 1: Pull
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for BGA contact: GPIO1_9. 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up

Table A-1038. Register IOMUXC_SW_PAD_CTL_PAD_GPIO1_9 Bits Description

Field	Description
3 ODE	Open Drain Enable Field Select one out of next values for BGA contact: GPIO1_9. 0: Open Drain Disabled 1: Open Drain Enabled
2-1 DSE	Drive Strength Field Select one out of next values for BGA contact: GPIO1_9. 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0 SRE	Slew Rate Field Select one out of next values for BGA contact: GPIO1_9. 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1039. Register: IOMUXC_SW_PAD_CTL_GRP_CSI2_PKE0

Offset: 0x081c (IOMUXC_SW_PAD_CTL_GRP_CSI2_PKE0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1040. Register IOMUXC_SW_PAD_CTL_GRP_CSI2_PKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: CSI2_PKE0 (Pads: CSI2_D14 CSI2_D15 CSI2_D16 CSI2_D17). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1041. Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKS

Offset	0x0820 (IOMUXC_SW_PAD_CTL_GRP_DDRPKS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1042. Register IOMUXC_SW_PAD_CTL_GRP_DDRPKS Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for group: DDRPKS (Pads: DRAM_D0 DRAM_D1 DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D2 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23 DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D3 DRAM_D30 DRAM_D31 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7 DRAM_D8 DRAM_D9). 0: Keeper 1: Pull
5-0	Reserved

Table A-1043. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_SR1

Offset: 0x0824 (IOMUXC_SW_PAD_CTL_GRP_EIM_SR1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
W																SRE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1044. Register IOMUXC_SW_PAD_CTL_GRP_EIM_SR1 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EIM_SR1 (Pads: EIM_DA0 EIM_DA1 EIM_DA2 EIM_DA3). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1045. Register: IOMUXC_SW_PAD_CTL_GRP_DISP2_PKE0

Offset	0x0828 (IOMUXC_SW_PAD_CTL_GRP_DISP2_PKE0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1046. Register IOMUXC_SW_PAD_CTL_GRP_DISP2_PKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: DISP2_PKE0 (Pads: DISP2_DAT2 DISP2_DAT3 DISP2_DAT4 DISP2_DAT5). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1047. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_B4

Offset: 0x082c (IOMUXC_SW_PAD_CTL_GRP_DRAM_B4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1048. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_B4 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DRAM_B4 (Pads: DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D30 DRAM_D31). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1049. Register: IOMUXC_SW_PAD_CTL_GRP_INDDR

Offset: 0x0830 (IOMUXC_SW_PAD_CTL_GRP_INDDR) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DDR_INPUT	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1050. Register IOMUXC_SW_PAD_CTL_GRP_INDDR Bits Description

Field	Description
31-10	Reserved
9 DDR_INPUT	DDR / CMOS Input Mode Field. Select one out of next values for group: INDDR (Pads: DRAM_A0 DRAM_A1 DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7 DRAM_A8 DRAM_A9 DRAM_CAS DRAM_CS0 DRAM_CS1 DRAM_RAS DRAM_SDCKE0 DRAM_SDCKE1 DRAM_SDWE EIM_SDBA0 EIM_SDBA1). 0: CMOS input type 1: DDR2 input type
8-0	Reserved

Table A-1051. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_SR2

Offset: 0x0834 (IOMUXC_SW_PAD_CTL_GRP_EIM_SR2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1052. Register IOMUXC_SW_PAD_CTL_GRP_EIM_SR2 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EIM_SR2 (Pads: EIM_DA4 EIM_DA5 EIM_DA6 EIM_DA7). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1053. Register: IOMUXC_SW_PAD_CTL_GRP_PKEDDR

Offset: 0x0838 (IOMUXC_SW_PAD_CTL_GRP_PKEDDR) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1054. Register IOMUXC_SW_PAD_CTL_GRP_PKEDDR Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: PKEDDR (Pads: DRAM_D0 DRAM_D1 DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D2 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23 DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D3 DRAM_D30 DRAM_D31 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7 DRAM_D8 DRAM_D9). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1055. Register: IOMUXC_SW_PAD_CTL_GRP_DDR_A0

Offset: 0x083c (IOMUXC_SW_PAD_CTL_GRP_DDR_A0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1056. Register IOMUXC_SW_PAD_CTL_GRP_DDR_A0 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DDR_A0 (Pads: DRAM_A0 DRAM_A1 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1057. Register: IOMUXC_SW_PAD_CTL_GRP_EMI_PKE0

Offset: 0x0840 (IOMUXC_SW_PAD_CTL_GRP_EMI_PKE0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1058. Register IOMUXC_SW_PAD_CTL_GRP_EMI_PKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: EMI_PKE0 (Pads: EIM_DA0 EIM_DA1 EIM_DA10 EIM_DA11 EIM_DA12 EIM_DA13 EIM_DA14 EIM_DA15 EIM_DA2 EIM_DA3 EIM_DA4 EIM_DA5 EIM_DA6 EIM_DA7 EIM_DA8 EIM_DA9). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1059. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_SR3

Offset: 0x0844 (IOMUXC_SW_PAD_CTL_GRP_EIM_SR3) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1060. Register IOMUXC_SW_PAD_CTL_GRP_EIM_SR3 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EIM_SR3 (Pads: EIM_DA10 EIM_DA11 EIM_DA8 EIM_DA9). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1061. Register: IOMUXC_SW_PAD_CTL_GRP_DDR_A1

Offset: 0x0848 (IOMUXC_SW_PAD_CTL_GRP_DDR_A1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1062. Register IOMUXC_SW_PAD_CTL_GRP_DDR_A1 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DDR_A1 (Pads: DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A8 DRAM_A9 EIM_SDBA0 EIM_SDBA1 EIM_SDBA2). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1063. Register: IOMUXC_SW_PAD_CTL_GRP_DDRAPUS

Offset	0x084c (IOMUXC_SW_PAD_CTL_GRP_DDRAPUS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	PUS		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table A-1064. Register IOMUXC_SW_PAD_CTL_GRP_DDRAPUS Bits Description

Field	Description
31-6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for group: DDRAPUS (Pads: DRAM_A0 DRAM_A1 DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7 DRAM_A8 DRAM_A9 DRAM_CAS DRAM_RAS EIM_SDBA0 EIM_SDBA1). 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-1065. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_SR4

Offset	0x0850 (IOMUXC_SW_PAD_CTL_GRP_EIM_SR4)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1066. Register IOMUXC_SW_PAD_CTL_GRP_EIM_SR4 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EIM_SR4 (Pads: EIM_DA12 EIM_DA13 EIM_DA14 EIM_DA15). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1067. Register: IOMUXC_SW_PAD_CTL_GRP_EMI_SR5

Offset: 0x0854 (IOMUXC_SW_PAD_CTL_GRP_EMI_SR5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1068. Register IOMUXC_SW_PAD_CTL_GRP_EMI_SR5 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EMI_SR5 (Pads: EIM_A16 EIM_A17 EIM_A18 EIM_A19 EIM_A20 EIM_A21 EIM_A22 EIM_A23). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1069. Register: IOMUXC_SW_PAD_CTL_GRP_EMI_SR6

Offset	0x0858 (IOMUXC_SW_PAD_CTL_GRP_EMI_SR6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1070. Register IOMUXC_SW_PAD_CTL_GRP_EMI_SR6 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: EMI_SR6 (Pads: EIM_A24 EIM_A25 EIM_A26 EIM_A27). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1071. Register: IOMUXC_SW_PAD_CTL_GRP_HYSDDR0

Offset	0x085c (IOMUXC_SW_PAD_CTL_GRP_HYSDDR0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1072. Register IOMUXC_SW_PAD_CTL_GRP_HYSDDR0 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for group: HYSDDR0 (Pads: DRAM_D0 DRAM_D1 DRAM_D2 DRAM_D3 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7). 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-1073. Register: IOMUXC_SW_PAD_CTL_GRP_CSI1_PKE0

Offset: 0x0860 (IOMUXC_SW_PAD_CTL_GRP_CSI1_PKE0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1074. Register IOMUXC_SW_PAD_CTL_GRP_CSI1_PKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: CSI1_PKE0 (Pads: CSI1_D10 CSI1_D11 CSI1_D12 CSI1_D13 CSI1_D14 CSI1_D15 CSI1_D16 CSI1_D17 CSI1_D18 CSI1_D19). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1075. Register: IOMUXC_SW_PAD_CTL_GRP_HYSDDR1

Offset	0x0864 (IOMUXC_SW_PAD_CTL_GRP_HYSDDR1)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1076. Register IOMUXC_SW_PAD_CTL_GRP_HYSDDR1 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for group: HYSDDR1 (Pads: DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D8 DRAM_D9). 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-1077. Register: IOMUXC_SW_PAD_CTL_GRP_DISP1_PKE0

Offset	0x0868 (IOMUXC_SW_PAD_CTL_GRP_DISP1_PKE0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1078. Register IOMUXC_SW_PAD_CTL_GRP_DISP1_PKE0 Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: DISP1_PKE0 (Pads: DISP1_DAT0 DISP1_DAT1 DISP1_DAT2 DISP1_DAT3 DISP1_DAT4 DISP1_DAT5). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1079. Register: IOMUXC_SW_PAD_CTL_GRP_HYSDDR2

Offset	0x086c (IOMUXC_SW_PAD_CTL_GRP_HYSDDR2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1080. Register IOMUXC_SW_PAD_CTL_GRP_HYSDDR2 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for group: HYSDDR2 (Pads: DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23). 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-1081. Register: IOMUXC_SW_PAD_CTL_GRP_HVDDR

Offset	0x0870 (IOMUXC_SW_PAD_CTL_GRP_HVDDR)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	HVE	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1082. Register IOMUXC_SW_PAD_CTL_GRP_HVDDR Bits Description

Field	Description
31-14	Reserved
13 HVE	High / Low Output Voltage Range. This bit selects the output voltage mode for group: HVDDR (Pads: DRAM_A0 DRAM_A1 DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7 DRAM_A8 DRAM_A9 DRAM_CAS DRAM_CS0 DRAM_CS1 DRAM_D0 DRAM_D1 DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D2 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23 DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D3 DRAM_D30 DRAM_D31 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7 DRAM_D8 DRAM_D9 DRAM_DQM0 DRAM_DQM1 DRAM_DQM2 DRAM_DQM3 DRAM_RAS DRAM_SDCKE0 DRAM_SDCKE1 DRAM_SDCLK DRAM_SDQS0 DRAM_SDQS1 DRAM_SDQS2 DRAM_SDQS3 DRAM_SDWE EIM_SDBA0 EIM_SDBA1 EIM_SDBA2 EIM_SDODT0 EIM_SDODT1). 0 High output voltage mode 1 Low output voltage mode
12-0	Reserved

Table A-1083. Register: IOMUXC_SW_PAD_CTL_GRP_HYSDDR3

Offset	0x0874 (IOMUXC_SW_PAD_CTL_GRP_HYSDDR3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	HYS	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1084. Register IOMUXC_SW_PAD_CTL_GRP_HYSDDR3 Bits Description

Field	Description
31-9	Reserved
8 HYS	Hyst. Enable Field Select one out of next values for group: HYSDDR3 (Pads: DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D30 DRAM_D31). 0: Hysteresis Disabled 1: Hysteresis Enabled
7-0	Reserved

Table A-1085. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B0

Offset: 0x0878 (IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1086. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B0 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DRAM_SR_B0 (Pads: DRAM_D0 DRAM_D1 DRAM_D2 DRAM_D3 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1087. Register: IOMUXC_SW_PAD_CTL_GRP_DDRAPKS

Offset	0x087c (IOMUXC_SW_PAD_CTL_GRP_DDRAPKS)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	PUE	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Table A-1088. Register IOMUXC_SW_PAD_CTL_GRP_DDRAPKS Bits Description

Field	Description
31-7	Reserved
6 PUE	Pull / Keep Select Field Select one out of next values for group: DDRAPKS (Pads: DRAM_A0 DRAM_A1 DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7 DRAM_A8 DRAM_A9 DRAM_CAS DRAM_RAS EIM_SDBA0 EIM_SDBA1). 0: Keeper 1: Pull
5-0	Reserved

Table A-1089. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B1

Offset: 0x0880 (IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1090. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B1 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DRAM_SR_B1 (Pads: DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D8 DRAM_D9). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1091. Register: IOMUXC_SW_PAD_CTL_GRP_DDRPUS

Offset: 0x0884 (IOMUXC_SW_PAD_CTL_GRP_DDRPUS) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	PUS		0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Table A-1092. Register IOMUXC_SW_PAD_CTL_GRP_DDRPUS Bits Description

Field	Description
31-6	Reserved
5-4 PUS	Pull Up / Down Config. Field Select one out of next values for group: DDRPUS (Pads: DRAM_D0 DRAM_D1 DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D2 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23 DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D3 DRAM_D30 DRAM_D31 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7 DRAM_D8 DRAM_D9). 00: 100KOhm Pull Down 01: 47KOhm Pull Up 10: 100KOhm Pull Up 11: 22KOhm Pull Up
3-0	Reserved

Table A-1093. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_DS1

Offset: 0x0888 (IOMUXC_SW_PAD_CTL_GRP_EIM_DS1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1094. Register IOMUXC_SW_PAD_CTL_GRP_EIM_DS1 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EIM_DS1 (Pads: EIM_DA0 EIM_DA1 EIM_DA2 EIM_DA3). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1095. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B2

Offset	0x088c (IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B2)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1096. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B2 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DRAM_SR_B2 (Pads: DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1097. Register: IOMUXC_SW_PAD_CTL_GRP_PKEADDR

Offset	0x0890 (IOMUXC_SW_PAD_CTL_GRP_PKEADDR)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	PKE	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Table A-1098. Register IOMUXC_SW_PAD_CTL_GRP_PKEADDR Bits Description

Field	Description
31-8	Reserved
7 PKE	Pull / Keep Enable Field Select one out of next values for group: PKEADDR (Pads: DRAM_A0 DRAM_A1 DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7 DRAM_A8 DRAM_A9 DRAM_CAS DRAM_RAS EIM_SDBA0 EIM_SDBA1). 0: Pull/Keeper Disabled 1: Pull/Keeper Enabled
6-0	Reserved

Table A-1099. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_DS2

Offset: 0x0894 (IOMUXC_SW_PAD_CTL_GRP_EIM_DS2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1100. Register IOMUXC_SW_PAD_CTL_GRP_EIM_DS2 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EIM_DS2 (Pads: EIM_DA4 EIM_DA5 EIM_DA6 EIM_DA7). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1101. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_DS3

Offset	0x0898 (IOMUXC_SW_PAD_CTL_GRP_EIM_DS3)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1102. Register IOMUXC_SW_PAD_CTL_GRP_EIM_DS3 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EIM_DS3 (Pads: EIM_DA10 EIM_DA11 EIM_DA8 EIM_DA9). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1103. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B4

Offset: 0x089c (IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1104. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_SR_B4 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DRAM_SR_B4 (Pads: DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D30 DRAM_D31). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1105. Register: IOMUXC_SW_PAD_CTL_GRP_INMODE1

Offset: 0x08a0 (IOMUXC_SW_PAD_CTL_GRP_INMODE1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	DDR_INPUT	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1106. Register IOMUXC_SW_PAD_CTL_GRP_INMODE1 Bits Description

Field	Description
31-10	Reserved
9 DDR_INPUT	DDR / CMOS Input Mode Field Select one out of next values for group: INMODE1 (Pads: DRAM_D0 DRAM_D1 DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D2 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23 DRAM_D24 DRAM_D25 DRAM_D26 DRAM_D27 DRAM_D28 DRAM_D29 DRAM_D3 DRAM_D30 DRAM_D31 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7 DRAM_D8 DRAM_D9 DRAM_DQM0 DRAM_DQM1 DRAM_DQM2 DRAM_DQM3). 0: CMOS input type 1: DDR2 input type
8-0	Reserved

Table A-1107. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_B0

Offset: 0x08a4 (IOMUXC_SW_PAD_CTL_GRP_DRAM_B0) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1108. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_B0 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DRAM_B0 (Pads: DRAM_D0 DRAM_D1 DRAM_D2 DRAM_D3 DRAM_D4 DRAM_D5 DRAM_D6 DRAM_D7). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1109. Register: IOMUXC_SW_PAD_CTL_GRP_EIM_DS4

Offset: 0x08a8 (IOMUXC_SW_PAD_CTL_GRP_EIM_DS4) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1110. Register IOMUXC_SW_PAD_CTL_GRP_EIM_DS4 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EIM_DS4 (Pads: EIM_DA12 EIM_DA13 EIM_DA14 EIM_DA15). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1111. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_B1

Offset: 0x08ac (IOMUXC_SW_PAD_CTL_GRP_DRAM_B1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1112. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_B1 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DRAM_B1 (Pads: DRAM_D10 DRAM_D11 DRAM_D12 DRAM_D13 DRAM_D14 DRAM_D15 DRAM_D8 DRAM_D9). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1113. Register: IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A0

Offset	0x08b0 (IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A0)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1114. Register IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A0 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DDR_SR_A0 (Pads: DRAM_A0 DRAM_A1 DRAM_A2 DRAM_A3 DRAM_A4 DRAM_A5 DRAM_A6 DRAM_A7). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1115. Register: IOMUXC_SW_PAD_CTL_GRP_EMI_DS5

Offset: 0x08b4 (IOMUXC_SW_PAD_CTL_GRP_EMI_DS5) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1116. Register IOMUXC_SW_PAD_CTL_GRP_EMI_DS5 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EMI_DS5 (Pads: EIM_A16 EIM_A17 EIM_A18 EIM_A19 EIM_A20 EIM_A21 EIM_A22 EIM_A23). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1117. Register: IOMUXC_SW_PAD_CTL_GRP_DRAM_B2

Offset: 0x08b8 (IOMUXC_SW_PAD_CTL_GRP_DRAM_B2) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1118. Register IOMUXC_SW_PAD_CTL_GRP_DRAM_B2 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: DRAM_B2 (Pads: DRAM_D16 DRAM_D17 DRAM_D18 DRAM_D19 DRAM_D20 DRAM_D21 DRAM_D22 DRAM_D23). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1119. Register: IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A1

Offset: 0x08bc (IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A1) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SRE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table A-1120. Register IOMUXC_SW_PAD_CTL_GRP_DDR_SR_A1 Bits Description

Field	Description
31-1	Reserved
0 SRE	Slew Rate Field Select one out of next values for group: DDR_SR_A1 (Pads: DRAM_A10 DRAM_A11 DRAM_A12 DRAM_A13 DRAM_A14 DRAM_A8 DRAM_A9 EIM_SDBA0 EIM_SDBA1 EIM_SDBA2). 0: Slow Slew Rate 1: Fast Slew Rate

Table A-1121. Register: IOMUXC_SW_PAD_CTL_GRP_EMI_DS6

Offset	0x08c0 (IOMUXC_SW_PAD_CTL_GRP_EMI_DS6)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DSE		0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table A-1122. Register IOMUXC_SW_PAD_CTL_GRP_EMI_DS6 Bits Description

Field	Description
31-3	Reserved
2-1 DSE	Drive Strength Field Select one out of next values for group: EMI_DS6 (Pads: EIM_A24 EIM_A25 EIM_A26 EIM_A27). 00: Low Drive Strength 01: Medium Drive Strength 10: High Drive Strength 11: Max Drive Strength
0	Reserved

Table A-1123. Register: IOMUXC_AUDMUX_P4_INPUT_DA_AMX_SELECT_INPUT

Offset: 0x08c4 (IOMUXC_AUDMUX_P4_INPUT_DA_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1124. Register IOMUXC_AUDMUX_P4_INPUT_DA_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p4_input_da_amx 0: Selecting BGA contact: EIM_D21 for Mode: ALT5. 1: Selecting BGA contact: CSPI1_MISO for Mode: ALT1.

Table A-1125. Register: IOMUXC_AUDMUX_P4_INPUT_DB_AMX_SELECT_INPUT

Offset	0x08c8 (IOMUXC_AUDMUX_P4_INPUT_DB_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1126. Register IOMUXC_AUDMUX_P4_INPUT_DB_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p4_input_db_amx 0: Selecting BGA contact: EIM_D20 for Mode: ALT5. 1: Selecting BGA contact: CSPI1_SS1 for Mode: ALT1.

Table A-1127. Register: IOMUXC_AUDMUX_P4_INPUT_TXCLK_AMX_SELECT_INPUT

Offset	0x08cc (IOMUXC_AUDMUX_P4_INPUT_TXCLK_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1128. Register IOMUXC_AUDMUX_P4_INPUT_TXCLK_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p4_input_txclk_amx 0: Selecting BGA contact: EIM_D22 for Mode: ALT5. 1: Selecting BGA contact: CSPI1_SS0 for Mode: ALT1.

Table A-1129. Register: IOMUXC_AUDMUX_P4_INPUT_TXFS_AMX_SELECT_INPUT

Offset: 0x08d0 (IOMUXC_AUDMUX_P4_INPUT_TXFS_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1130. Register IOMUXC_AUDMUX_P4_INPUT_TXFS_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p4_input_txfs_amx 0: Selecting BGA contact: EIM_D23 for Mode: ALT5. 1: Selecting BGA contact: CSPI1_RDY for Mode: ALT1.

Table A-1131. Register: IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT

Offset: 0x08d4 (IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1132. Register IOMUXC_AUDMUX_P5_INPUT_DA_AMX_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_da_amx 00: Selecting BGA contact: EIM_D17 for Mode: ALT7. 01: Selecting BGA contact: EIM_CS3 for Mode: ALT6. 10: Selecting BGA contact: SD1_DATA1 for Mode: ALT1.

Table A-1133. Register: IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT

Offset	0x08d8 (IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1134. Register IOMUXC_AUDMUX_P5_INPUT_DB_AMX_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_db_amx 00: Selecting BGA contact: EIM_D16 for Mode: ALT7. 01: Selecting BGA contact: EIM_CS2 for Mode: ALT6. 10: Selecting BGA contact: SD1_DATA0 for Mode: ALT1.

Table A-1135. Register: IOMUXC_AUDMUX_P5_INPUT_RXCLK_AMX_SELECT_INPUT

Offset	0x08dc (IOMUXC_AUDMUX_P5_INPUT_RXCLK_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1136. Register IOMUXC_AUDMUX_P5_INPUT_RXCLK_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_rxclk_amx 0: Selecting BGA contact: EIM_EB3 for Mode: ALT6. 1: Selecting BGA contact: SD1_CLK for Mode: ALT1.

Table A-1137. Register: IOMUXC_AUDMUX_P5_INPUT_RXFS_AMX_SELECT_INPUT

Offset	0x08e0 (IOMUXC_AUDMUX_P5_INPUT_RXFS_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1138. Register IOMUXC_AUDMUX_P5_INPUT_RXFS_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_rxfs_amx 0: Selecting BGA contact: EIM_EB2 for Mode: ALT6. 1: Selecting BGA contact: SD1_CMD for Mode: ALT1.

Table A-1139. Register: IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT

Offset	0x08e4 (IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1140. Register IOMUXC_AUDMUX_P5_INPUT_TXCLK_AMX_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_txclk_amx 00: Selecting BGA contact: EIM_D18 for Mode: ALT7. 01: Selecting BGA contact: EIM_CS4 for Mode: ALT6. 10: Selecting BGA contact: SD1_DATA2 for Mode: ALT1.

Table A-1141. Register: IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT

Offset: 0x08e8 (IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1142. Register IOMUXC_AUDMUX_P5_INPUT_TXFS_AMX_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p5_input_txfs_amx 00: Selecting BGA contact: EIM_D19 for Mode: ALT7. 01: Selecting BGA contact: EIM_CS5 for Mode: ALT6. 10: Selecting BGA contact: SD1_DATA3 for Mode: ALT1.

Table A-1143. Register: IOMUXC_AUDMUX_P6_INPUT_DA_AMX_SELECT_INPUT

Offset	0x08ec (IOMUXC_AUDMUX_P6_INPUT_DA_AMX_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1144. Register IOMUXC_AUDMUX_P6_INPUT_DA_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_da_amx 0: Selecting BGA contact: EIM_D29 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT12 for Mode: ALT4.

Table A-1145. Register: IOMUXC_AUDMUX_P6_INPUT_DB_AMX_SELECT_INPUT

Offset	0x08f0 (IOMUXC_AUDMUX_P6_INPUT_DB_AMX_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1146. Register IOMUXC_AUDMUX_P6_INPUT_DB_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_db_amx 0: Selecting BGA contact: EIM_D28 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT11 for Mode: ALT4.

Table A-1147. Register: IOMUXC_AUDMUX_P6_INPUT_RXCLK_AMX_SELECT_INPUT

Offset	0x08f4 (IOMUXC_AUDMUX_P6_INPUT_RXCLK_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1148. Register IOMUXC_AUDMUX_P6_INPUT_RXCLK_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_rxclk_amx 0: Selecting BGA contact: EIM_D27 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT9 for Mode: ALT4.

Table A-1149. Register: IOMUXC_AUDMUX_P6_INPUT_RXFS_AMX_SELECT_INPUT

Offset	0x08f8 (IOMUXC_AUDMUX_P6_INPUT_RXFS_AMX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1150. Register IOMUXC_AUDMUX_P6_INPUT_RXFS_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_rxfs_amx 0: Selecting BGA contact: EIM_D24 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT15 for Mode: ALT4.

Table A-1151. Register: IOMUXC_AUDMUX_P6_INPUT_TXCLK_AMX_SELECT_INPUT

Offset: 0x08fc (IOMUXC_AUDMUX_P6_INPUT_TXCLK_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1152. Register IOMUXC_AUDMUX_P6_INPUT_TXCLK_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_txclk_amx 0: Selecting BGA contact: EIM_D30 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT13 for Mode: ALT4.

Table A-1153. Register: IOMUXC_AUDMUX_P6_INPUT_TXFS_AMX_SELECT_INPUT

Offset: 0x0900 (IOMUXC_AUDMUX_P6_INPUT_TXFS_AMX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1154. Register IOMUXC_AUDMUX_P6_INPUT_TXFS_AMX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: audmux, In Pin: p6_input_txfs_amx 0: Selecting BGA contact: EIM_D31 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT14 for Mode: ALT4.

Table A-1155. Register: IOMUXC_CCM_IPP_DI0_CLK_SELECT_INPUT

Offset: 0x0904 (IOMUXC_CCM_IPP_DI0_CLK_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1156. Register IOMUXC_CCM_IPP_DI0_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ccm, In Pin: ipp_di0_clk 0: Selecting BGA contact: EIM_CS5 for Mode: ALT4. 1: Selecting BGA contact: DI_GP1 for Mode: ALT2.

Table A-1157. Register: IOMUXC_CCM_IPP_DI1_CLK_SELECT_INPUT

Offset: 0x0908 (IOMUXC_CCM_IPP_DI1_CLK_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1158. Register IOMUXC_CCM_IPP_DI1_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ccm, In Pin: ipp_di1_clk 0: Selecting BGA contact: EIM_A26 for Mode: ALT6. 1: Selecting BGA contact: GPIO1_4 for Mode: ALT4.

Table A-1159. Register: IOMUXC_CCM_PLL1_BYPASS_CLK_SELECT_INPUT

Offset: 0x090c (IOMUXC_CCM_PLL1_BYPASS_CLK_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1160. Register IOMUXC_CCM_PLL1_BYPASS_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ccm, In Pin: pll1_bypass_clk 0: Selecting BGA contact: KEY_COL0 for Mode: ALT7. 1: Selecting BGA contact: GPIO1_2 for Mode: ALT7.

Table A-1161. Register: IOMUXC_CCM_PLL2_BYPASS_CLK_SELECT_INPUT

Offset: 0x0910 (IOMUXC_CCM_PLL2_BYPASS_CLK_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1162. Register IOMUXC_CCM_PLL2_BYPASS_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ccm, In Pin: pll2_bypass_clk 0: Selecting BGA contact: KEY_COL1 for Mode: ALT7. 1: Selecting BGA contact: GPIO1_3 for Mode: ALT7.

Table A-1163. Register: IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT

Offset: 0x0914 (IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1164. Register IOMUXC_CSPI_IPP_CSPI_CLK_IN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_cspi_clk_in 00: Selecting BGA contact: NANDF_CS2 for Mode: ALT6. 01: Selecting BGA contact: USBH1_CLK for Mode: ALT1. 10: Selecting BGA contact: SD1_CLK for Mode: ALT2. 11: Selecting BGA contact: SD2_CLK for Mode: ALT2.

Table A-1165. Register: IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT

Offset 0x0918 (IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1166. Register IOMUXC_CSPI_IPP_IND_MISO_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_ind_miso 00: Selecting BGA contact: USBH1_NXT for Mode: ALT1. 01: Selecting BGA contact: SD1_DATA0 for Mode: ALT2. 10: Selecting BGA contact: GPIO1_1 for Mode: ALT2. 11: Selecting BGA contact: SD2_DATA0 for Mode: ALT2.

Table A-1167. Register: IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT

Offset 0x091c (IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1168. Register IOMUXC_CSPI_IPP_IND_MOSI_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_ind_mosi 00: Selecting BGA contact: NANDF_RB1 for Mode: ALT6. 01: Selecting BGA contact: USBH1_DIR for Mode: ALT1. 10: Selecting BGA contact: SD1_CMD for Mode: ALT2. 11: Selecting BGA contact: SD2_CMD for Mode: ALT2.

Table A-1169. Register: IOMUXC_CSPI_IPP_IND_SS1_B_SELECT_INPUT

Offset: 0x0920 (IOMUXC_CSPI_IPP_IND_SS1_B_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1170. Register IOMUXC_CSPI_IPP_IND_SS1_B_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_ind_ss1_b 0: Selecting BGA contact: USBH1_DATA5 for Mode: ALT1. 1: Selecting BGA contact: SD1_DATA3 for Mode: ALT2.

Table A-1171. Register: IOMUXC_CSPI_IPP_IND_SS2_B_SELECT_INPUT

Offset: 0x0924 (IOMUXC_CSPI_IPP_IND_SS2_B_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1172. Register IOMUXC_CSPI_IPP_IND_SS2_B_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_ind_ss2_b 0: Selecting BGA contact: GPIO1_0 for Mode: ALT2. 1: Selecting BGA contact: SD2_DATA3 for Mode: ALT2.

Table A-1173. Register: IOMUXC_CSPI_IPP_IND_SS3_B_SELECT_INPUT

Offset: 0x0928 (IOMUXC_CSPI_IPP_IND_SS3_B_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1174. Register IOMUXC_CSPI_IPP_IND_SS3_B_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: cspi, In Pin: ipp_ind_ss3_b 0: Selecting BGA contact: NANDF_CS6 for Mode: ALT7. 1: Selecting BGA contact: USBH1_DATA6 for Mode: ALT1.

Table A-1175. Register: IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT

Offset 0x092c (IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1176. Register IOMUXC_DPLLIP1_L1T_TOG_EN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: dpllip1, In Pin: l1t_tog_en 00: Selecting BGA contact: NANDF_RB3 for Mode: ALT4. 01: Selecting BGA contact: GPIO1_2 for Mode: ALT6. 10: Selecting BGA contact: GPIO1_4 for Mode: ALT7. 11: Selecting BGA contact: GPIO1_7 for Mode: ALT7.

Table A-1177. Register: IOMUXC_ECSPI2_IPP_IND_SS_B_1_SELECT_INPUT

Offset: 0x0930 (IOMUXC_ECSPI2_IPP_IND_SS_B_1_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1178. Register IOMUXC_ECSPI2_IPP_IND_SS_B_1_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecspi2, In Pin: ipp_ind_ss_b[1] 0: Selecting BGA contact: NANDF_RB0 for Mode: ALT5. 1: Selecting BGA contact: NANDF_D12 for Mode: ALT2.

Table A-1179. Register: IOMUXC_ECSPI2_IPP_IND_SS_B_3_SELECT_INPUT

Offset: 0x0934 (IOMUXC_ECSPI2_IPP_IND_SS_B_3_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1180. Register IOMUXC_ECSPI2_IPP_IND_SS_B_3_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ecspi2, In Pin: ipp_ind_ss_b[3] 0: Selecting BGA contact: NANDF_D14 for Mode: ALT2. 1: Selecting BGA contact: USBH1_DATA7 for Mode: ALT5.

Table A-1181. Register: IOMUXC_EMI_IPP_IND_RDY_INT_SELECT_INPUT

Offset: 0x0938 (IOMUXC_EMI_IPP_IND_RDY_INT_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1182. Register IOMUXC_EMI_IPP_IND_RDY_INT_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: emi, In Pin: ipp_ind_rdy_int 0: Selecting BGA contact: NANDF_RDY_INT for Mode: ALT0. 1: Selecting BGA contact: GPIO1_4 for Mode: ALT3.

Table A-1183. Register: IOMUXC_ESDHC3_IPP_DAT0_IN_SELECT_INPUT

Offset: 0x093c (IOMUXC_ESDHC3_IPP_DAT0_IN_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1184. Register IOMUXC_ESDHC3_IPP_DAT0_IN_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: esdhc3, In Pin: ipp_dat0_in 0: Selecting BGA contact: NANDF_WE_B for Mode: ALT2. 1: Selecting BGA contact: NANDF_D8 for Mode: ALT5.

Table A-1185. Register: IOMUXC_ESDHC3_IPP_DAT1_IN_SELECT_INPUT

Offset	0x0940 (IOMUXC_ESDHC3_IPP_DAT1_IN_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1186. Register IOMUXC_ESDHC3_IPP_DAT1_IN_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: esdhc3, In Pin: ipp_dat1_in 0: Selecting BGA contact: NANDF_RE_B for Mode: ALT2. 1: Selecting BGA contact: NANDF_D9 for Mode: ALT5.

Table A-1187. Register: IOMUXC_ESDHC3_IPP_DAT2_IN_SELECT_INPUT

Offset: 0x0944 (IOMUXC_ESDHC3_IPP_DAT2_IN_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1188. Register IOMUXC_ESDHC3_IPP_DAT2_IN_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: esdhc3, In Pin: ipp_dat2_in 0: Selecting BGA contact: NANDF_WP_B for Mode: ALT2. 1: Selecting BGA contact: NANDF_D10 for Mode: ALT5.

Table A-1189. Register: IOMUXC_ESDHC3_IPP_DAT3_IN_SELECT_INPUT

Offset: 0x0948 (IOMUXC_ESDHC3_IPP_DAT3_IN_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1190. Register IOMUXC_ESDHC3_IPP_DAT3_IN_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: esdhc3, In Pin: ipp_dat3_in 0: Selecting BGA contact: NANDF_RB0 for Mode: ALT2. 1: Selecting BGA contact: NANDF_D11 for Mode: ALT5.

Table A-1191. Register: IOMUXC_FEC_FEC_COL_SELECT_INPUT

Offset 0x094c (IOMUXC_FEC_FEC_COL_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1192. Register IOMUXC_FEC_FEC_COL_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_col 0: Selecting BGA contact: NANDF_RB2 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT10 for Mode: ALT2.

Table A-1193. Register: IOMUXC_FEC_FEC_CRS_SELECT_INPUT

Offset: 0x0950 (IOMUXC_FEC_FEC_CRS_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1194. Register IOMUXC_FEC_FEC_CRS_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_crs 0: Selecting BGA contact: EIM_CS5 for Mode: ALT3. 1: Selecting BGA contact: DI2_PIN4 for Mode: ALT2.

Table A-1195. Register: IOMUXC_FEC_FEC_MDI_SELECT_INPUT

Offset: 0x0954 (IOMUXC_FEC_FEC_MDI_SELECT_INPUT)

Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1196. Register IOMUXC_FEC_FEC_MDI_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_mdi 0: Selecting BGA contact: EIM_EB2 for Mode: ALT3. 1: Selecting BGA contact: DI2_PIN3 for Mode: ALT2.

Table A-1197. Register: IOMUXC_FEC_FEC_RDATA_0_SELECT_INPUT

Offset: 0x0958 (IOMUXC_FEC_FEC_RDATA_0_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1198. Register IOMUXC_FEC_FEC_RDATA_0_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rdata[0] 0: Selecting BGA contact: NANDF_D9 for Mode: ALT2. 1: Selecting BGA contact: DISP2_DAT14 for Mode: ALT2.

Table A-1199. Register: IOMUXC_FEC_FEC_RDATA_1_SELECT_INPUT

Offset: 0x095c (IOMUXC_FEC_FEC_RDATA_1_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1200. Register IOMUXC_FEC_FEC_RDATA_1_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rdata[1] 0: Selecting BGA contact: EIM_EB3 for Mode: ALT3. 1: Selecting BGA contact: DI2_DISP_CLK for Mode: ALT2.

Table A-1201. Register: IOMUXC_FEC_FEC_RDATA_2_SELECT_INPUT

Offset: 0x0960 (IOMUXC_FEC_FEC_RDATA_2_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1202. Register IOMUXC_FEC_FEC_RDATA_2_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rdata[2] 0: Selecting BGA contact: EIM_CS2 for Mode: ALT3. 1: Selecting BGA contact: DI_GP4 for Mode: ALT2.

Table A-1203. Register: IOMUXC_FEC_FEC_RDATA_3_SELECT_INPUT

Offset: 0x0964 (IOMUXC_FEC_FEC_RDATA_3_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1204. Register IOMUXC_FEC_FEC_RDATA_3_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rdata[3] 0: Selecting BGA contact: EIM_CS3 for Mode: ALT3. 1: Selecting BGA contact: DISP2_DAT0 for Mode: ALT2.

Table A-1205. Register: IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT

Offset: 0x0968 (IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1206. Register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rx_clk 0: Selecting BGA contact: NANDF_RB3 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT11 for Mode: ALT2.

Table A-1207. Register: IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT

Offset 0x096c (IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1208. Register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rx_dv 0: Selecting BGA contact: NANDF_D11 for Mode: ALT2. 1: Selecting BGA contact: DISP2_DAT12 for Mode: ALT2.

Table A-1209. Register: IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT

Offset: 0x0970 (IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1210. Register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_rx_er 0: Selecting BGA contact: EIM_CS4 for Mode: ALT3. 1: Selecting BGA contact: DISP2_DAT1 for Mode: ALT2.

Table A-1211. Register: IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT

Offset 0x0974 (IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1212. Register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: fec, In Pin: fec_tx_clk 0: Selecting BGA contact: NANDF_RDY_INT for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT13 for Mode: ALT2.

Table A-1213. Register: IOMUXC_GPIO3_IPP_IND_G_IN_1_SELECT_INPUT

Offset: 0x0978 (IOMUXC_GPIO3_IPP_IND_G_IN_1_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1214. Register IOMUXC_GPIO3_IPP_IND_G_IN_1_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[1] 0: Selecting BGA contact: EIM_LBA for Mode: ALT1. 1: Selecting BGA contact: DI1_PIN12 for Mode: ALT4.

Table A-1215. Register: IOMUXC_GPIO3_IPP_IND_G_IN_2_SELECT_INPUT

Offset 0x097c (IOMUXC_GPIO3_IPP_IND_G_IN_2_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1216. Register IOMUXC_GPIO3_IPP_IND_G_IN_2_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[2] 0: Selecting BGA contact: EIM_CRE for Mode: ALT1. 1: Selecting BGA contact: DI1_PIN13 for Mode: ALT4.

Table A-1217. Register: IOMUXC_GPIO3_IPP_IND_G_IN_3_SELECT_INPUT

Offset	0x0980 (IOMUXC_GPIO3_IPP_IND_G_IN_3_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1218. Register IOMUXC_GPIO3_IPP_IND_G_IN_3_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[3] 0: Selecting BGA contact: NANDF_WE_B for Mode: ALT3. 1: Selecting BGA contact: DI1_D0_CS for Mode: ALT4.

Table A-1219. Register: IOMUXC_GPIO3_IPP_IND_G_IN_4_SELECT_INPUT

Offset: 0x0984 (IOMUXC_GPIO3_IPP_IND_G_IN_4_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1220. Register IOMUXC_GPIO3_IPP_IND_G_IN_4_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[4] 0: Selecting BGA contact: NANDF_RE_B for Mode: ALT3. 1: Selecting BGA contact: DI1_D1_CS for Mode: ALT4.

Table A-1221. Register: IOMUXC_GPIO3_IPP_IND_G_IN_5_SELECT_INPUT

Offset	0x0988 (IOMUXC_GPIO3_IPP_IND_G_IN_5_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1222. Register IOMUXC_GPIO3_IPP_IND_G_IN_5_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[5] 0: Selecting BGA contact: NANDF_ALE for Mode: ALT3. 1: Selecting BGA contact: DISPB2_SER_DIN for Mode: ALT4.

Table A-1223. Register: IOMUXC_GPIO3_IPP_IND_G_IN_6_SELECT_INPUT

Offset: 0x098c (IOMUXC_GPIO3_IPP_IND_G_IN_6_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1224. Register IOMUXC_GPIO3_IPP_IND_G_IN_6_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[6] 0: Selecting BGA contact: NANDF_CLE for Mode: ALT3. 1: Selecting BGA contact: DISPB2_SER_DIO for Mode: ALT4.

Table A-1225. Register: IOMUXC_GPIO3_IPP_IND_G_IN_7_SELECT_INPUT

Offset	0x0990 (IOMUXC_GPIO3_IPP_IND_G_IN_7_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1226. Register IOMUXC_GPIO3_IPP_IND_G_IN_7_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[7] 0: Selecting BGA contact: NANDF_WP_B for Mode: ALT3. 1: Selecting BGA contact: DISPB2_SER_CLK for Mode: ALT4.

Table A-1227. Register: IOMUXC_GPIO3_IPP_IND_G_IN_8_SELECT_INPUT

Offset	0x0994 (IOMUXC_GPIO3_IPP_IND_G_IN_8_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1228. Register IOMUXC_GPIO3_IPP_IND_G_IN_8_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[8] 0: Selecting BGA contact: NANDF_RB0 for Mode: ALT3. 1: Selecting BGA contact: DISPB2_SER_RS for Mode: ALT4.

Table A-1229. Register: IOMUXC_GPIO3_IPP_IND_G_IN_12_SELECT_INPUT

Offset: 0x0998 (IOMUXC_GPIO3_IPP_IND_G_IN_12_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1230. Register IOMUXC_GPIO3_IPP_IND_G_IN_12_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: gpio3, In Pin: ipp_ind_g_in[12] 0: Selecting BGA contact: GPIO_NAND for Mode: ALT0. 1: Selecting BGA contact: CSI1_D8 for Mode: ALT3.

Table A-1231. Register: IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT

Offset	0x099c (IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1232. Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS1_DATA_EN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: hsc_mipi_mix, In Pin: ipp_ind_sens1_data_en 00: Selecting BGA contact: EIM_A27 for Mode: ALT5. 01: Selecting BGA contact: DI2_PIN4 for Mode: ALT3. 10: Selecting BGA contact: GPIO1_8 for Mode: ALT2.

Table A-1233. Register: IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS2_DATA_EN_SELECT_INPUT

Offset	0x09a0 (IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS2_DATA_EN_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1234. Register IOMUXC_HSC_MIPI_MIX_IPP_IND_SENS2_DATA_EN_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: hsc_mipi_mix, In Pin: ipp_ind_sens2_data_en 0: Selecting BGA contact: EIM_A26 for Mode: ALT5. 1: Selecting BGA contact: DI_GP3 for Mode: ALT3.

Table A-1235. Register: IOMUXC_HSC_MIPI_MIX_PAR0_VSYNC_SELECT_INPUT

Offset	0x09a4 (IOMUXC_HSC_MIPI_MIX_PAR0_VSYNC_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1236. Register IOMUXC_HSC_MIPI_MIX_PAR0_VSYNC_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: hsc_mipi_mix, In Pin: par0_vsync 0: Selecting BGA contact: EIM_A27 for Mode: ALT6. 1: Selecting BGA contact: DISPB2_SER_DIN for Mode: ALT2.

Table A-1237. Register: IOMUXC_HSC_MIPI_MIX_PAR1_DI_WAIT_SELECT_INPUT

Offset	0x09a8 (IOMUXC_HSC_MIPI_MIX_PAR1_DI_WAIT_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1238. Register IOMUXC_HSC_MIPI_MIX_PAR1_DI_WAIT_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: hsc_mipi_mix, In Pin: par1_di_wait 0: Selecting BGA contact: NANDF_RB2 for Mode: ALT5. 1: Selecting BGA contact: DI_GP2 for Mode: ALT2.

Table A-1239. Register: IOMUXC_HSC_MIPI_MIX_PAR_SISG_TRIG_SELECT_INPUT

Offset	0x09ac (IOMUXC_HSC_MIPI_MIX_PAR_SISG_TRIG_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1240. Register IOMUXC_HSC_MIPI_MIX_PAR_SISG_TRIG_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: hsc_mipi_mix, In Pin: par_sisg_trig 0: Selecting BGA contact: KEY_COL5 for Mode: ALT6. 1: Selecting BGA contact: GPIO1_6 for Mode: ALT2.

Table A-1241. Register: IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT

Offset 0x09b0 (IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1242. Register IOMUXC_I2C1_IPP_SCL_IN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i2c1, In Pin: ipp_scl_in 00: Selecting BGA contact: EIM_D19 for Mode: ALT4. 01: Selecting BGA contact: CSPI1_SCLK for Mode: ALT1. 10: Selecting BGA contact: SD2_CMD for Mode: ALT1.

Table A-1243. Register: IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT

Offset: 0x09b4 (IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1244. Register IOMUXC_I2C1_IPP_SDA_IN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i2c1, In Pin: ipp_sda_in 00: Selecting BGA contact: EIM_D16 for Mode: ALT4. 01: Selecting BGA contact: CSPI1_MOSI for Mode: ALT1. 10: Selecting BGA contact: SD2_CLK for Mode: ALT1.

Table A-1245. Register: IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT

Offset	0x09b8 (IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1246. Register IOMUXC_I2C2_IPP_SCL_IN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i2c2, In Pin: ipp_scl_in 00: Selecting BGA contact: EIM_D27 for Mode: ALT4. 01: Selecting BGA contact: KEY_COL4 for Mode: ALT3. 10: Selecting BGA contact: USBH1_CLK for Mode: ALT5. 11: Selecting BGA contact: GPIO1_2 for Mode: ALT2.

Table A-1247. Register: IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT

Offset	0x09bc (IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1248. Register IOMUXC_I2C2_IPP_SDA_IN_SELECT_INPUT Bits Description

Field	Description
31-2	Reserved
1-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: i2c2, In Pin: ipp_sda_in 00: Selecting BGA contact: EIM_D24 for Mode: ALT4. 01: Selecting BGA contact: KEY_COL5 for Mode: ALT3. 10: Selecting BGA contact: USBH1_DIR for Mode: ALT5. 11: Selecting BGA contact: GPIO1_3 for Mode: ALT2.

Table A-1249. Register: IOMUXC_IPU_IPP_DI_0_IND_DISPB_SD_D_SELECT_INPUT

Offset	0x09c0 (IOMUXC_IPU_IPP_DI_0_IND_DISPB_SD_D_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1250. Register IOMUXC_IPU_IPP_DI_0_IND_DISPB_SD_D_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ipu, In Pin: ipp_di_0_ind_dispb_sd_d 0: Selecting BGA contact: DI_GP3 for Mode: ALT0. 1: Selecting BGA contact: DI_GP4 for Mode: ALT0.

Table A-1251. Register: IOMUXC_IPU_IPP_DI_1_IND_DISPB_SD_D_SELECT_INPUT

Offset	0x09c4 (IOMUXC_IPU_IPP_DI_1_IND_DISPB_SD_D_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1252. Register IOMUXC_IPU_IPP_DI_1_IND_DISPB_SD_D_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: ipu, In Pin: ipp_di_1_ind_dispb_sd_d 0: Selecting BGA contact: DISPB2_SER_DIN for Mode: ALT0. 1: Selecting BGA contact: DISPB2_SER_DIO for Mode: ALT0.

Table A-1253. Register: IOMUXC_KPP_IPP_IND_COL_6_SELECT_INPUT

Offset: 0x09c8 (IOMUXC_KPP_IPP_IND_COL_6_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1254. Register IOMUXC_KPP_IPP_IND_COL_6_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col[6] 0: Selecting BGA contact: EIM_D25 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT0 for Mode: ALT4.

Table A-1255. Register: IOMUXC_KPP_IPP_IND_COL_7_SELECT_INPUT

Offset	0x09cc (IOMUXC_KPP_IPP_IND_COL_7_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1256. Register IOMUXC_KPP_IPP_IND_COL_7_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_col[7] 0: Selecting BGA contact: EIM_D26 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT1 for Mode: ALT4.

Table A-1257. Register: IOMUXC_KPP_IPP_IND_ROW_4_SELECT_INPUT

Offset: 0x09d0 (IOMUXC_KPP_IPP_IND_ROW_4_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1258. Register IOMUXC_KPP_IPP_IND_ROW_4_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row[4] 0: Selecting BGA contact: EIM_D28 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT6 for Mode: ALT4.

Table A-1259. Register: IOMUXC_KPP_IPP_IND_ROW_5_SELECT_INPUT

Offset	0x09d4 (IOMUXC_KPP_IPP_IND_ROW_5_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1260. Register IOMUXC_KPP_IPP_IND_ROW_5_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row[5] 0: Selecting BGA contact: EIM_D29 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT7 for Mode: ALT4.

Table A-1261. Register: IOMUXC_KPP_IPP_IND_ROW_6_SELECT_INPUT

Offset: 0x09d8 (IOMUXC_KPP_IPP_IND_ROW_6_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1262. Register IOMUXC_KPP_IPP_IND_ROW_6_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row[6] 0: Selecting BGA contact: EIM_D30 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT8 for Mode: ALT4.

Table A-1263. Register: IOMUXC_KPP_IPP_IND_ROW_7_SELECT_INPUT

Offset 0x09dc (IOMUXC_KPP_IPP_IND_ROW_7_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1264. Register IOMUXC_KPP_IPP_IND_ROW_7_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: kpp, In Pin: ipp_ind_row[7] 0: Selecting BGA contact: EIM_D31 for Mode: ALT1. 1: Selecting BGA contact: DISP2_DAT10 for Mode: ALT4.

Table A-1265. Register: IOMUXC_UART1_IPP_UART_RTS_B_SELECT_INPUT

Offset: 0x09e0 (IOMUXC_UART1_IPP_UART_RTS_B_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1266. Register IOMUXC_UART1_IPP_UART_RTS_B_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart1, In Pin: ipp_uart_rts_b 0: Selecting BGA contact: UART1_RTS for Mode: ALT0. 1: Selecting BGA contact: UART1_CTS for Mode: ALT0.

Table A-1267. Register: IOMUXC_UART1_IPP_UART_RXD_MUX_SELECT_INPUT

Offset: 0x09e4 (IOMUXC_UART1_IPP_UART_RXD_MUX_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1268. Register IOMUXC_UART1_IPP_UART_RXD_MUX_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart1, In Pin: ipp_uart_rxd_mux 0: Selecting BGA contact: UART1_RXD for Mode: ALT0. 1: Selecting BGA contact: UART1_TXD for Mode: ALT0.

Table A-1269. Register: IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT

Offset	0x09e8 (IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1270. Register IOMUXC_UART2_IPP_UART_RTS_B_SELECT_INPUT Bits Description

Field	Description
31-3	Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart2, In Pin: ipp_uart_rts_b 000: Selecting BGA contact: EIM_D16 for Mode: ALT3. 001: Selecting BGA contact: EIM_D19 for Mode: ALT3. 010: Selecting BGA contact: EIM_D25 for Mode: ALT4. 011: Selecting BGA contact: EIM_D26 for Mode: ALT4. 100: Selecting BGA contact: USBH1_DATA0 for Mode: ALT1. 101: Selecting BGA contact: USBH1_DATA3 for Mode: ALT1.

Table A-1271. Register: IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT

Offset	0x09ec (IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1272. Register IOMUXC_UART2_IPP_UART_RXD_MUX_SELECT_INPUT Bits Description

Field	Description
31-3	Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart2, In Pin: ipp_uart_rxd_mux 000: Selecting BGA contact: EIM_D17 for Mode: ALT3. 001: Selecting BGA contact: EIM_D18 for Mode: ALT3. 010: Selecting BGA contact: UART2_RXD for Mode: ALT0. 011: Selecting BGA contact: UART2_TXD for Mode: ALT0. 100: Selecting BGA contact: USBH1_DATA1 for Mode: ALT1. 101: Selecting BGA contact: USBH1_DATA2 for Mode: ALT1.

Table A-1273. Register: IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT

Offset: 0x09f0 (IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1274. Register IOMUXC_UART3_IPP_UART_RTS_B_SELECT_INPUT Bits Description

Field	Description
31-3	Reserved
2-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart3, In Pin: ipp_uart_rts_b 000: Selecting BGA contact: EIM_D17 for Mode: ALT4. 001: Selecting BGA contact: EIM_D18 for Mode: ALT4. 010: Selecting BGA contact: EIM_D24 for Mode: ALT3. 011: Selecting BGA contact: EIM_D27 for Mode: ALT3. 100: Selecting BGA contact: KEY_COL4 for Mode: ALT2. 101: Selecting BGA contact: KEY_COL5 for Mode: ALT2. 110: Selecting BGA contact: USBH1_CLK for Mode: ALT7. 111: Selecting BGA contact: USBH1_DIR for Mode: ALT7.

Table A-1275. Register: IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT

Offset	0x09f4 (IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	DAISY			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1276. Register IOMUXC_UART3_IPP_UART_RXD_MUX_SELECT_INPUT Bits Description

Field	Description
31-4	Reserved
3-0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: uart3, In Pin: ipp_uart_rxd_mux 0000: Selecting BGA contact: EIM_D25 for Mode: ALT3. 0001: Selecting BGA contact: EIM_D26 for Mode: ALT3. 0010: Selecting BGA contact: AUD3_BB_RXD for Mode: ALT1. 0011: Selecting BGA contact: AUD3_BB_FS for Mode: ALT1. 0100: Selecting BGA contact: UART3_RXD for Mode: ALT1. 0101: Selecting BGA contact: UART3_TXD for Mode: ALT1. 0110: Selecting BGA contact: USBH1_STP for Mode: ALT5. 0111: Selecting BGA contact: USBH1_NXT for Mode: ALT5. 1000: Selecting BGA contact: DISP2_DAT0 for Mode: ALT5. 1001: Selecting BGA contact: DISP2_DAT1 for Mode: ALT5.

Table A-1277. Register: IOMUXC_USBOH3_IPP_IND_UH3_CLK_SELECT_INPUT

Offset	0x09f8 (IOMUXC_USBOH3_IPP_IND_UH3_CLK_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1278. Register IOMUXC_USBOH3_IPP_IND_UH3_CLK_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_clk 0: Selecting BGA contact: NANDF_RB3 for Mode: ALT6. 1: Selecting BGA contact: DISP2_DAT0 for Mode: ALT3.

Table A-1279. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_0_SELECT_INPUT

Offset	0x09fc (IOMUXC_USBOH3_IPP_IND_UH3_DATA_0_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1280. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_0_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_0 0: Selecting BGA contact: NANDF_D7 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT8 for Mode: ALT3.

Table A-1281. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_1_SELECT_INPUT

Offset	0x0a00 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_1_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1282. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_1_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_1 0: Selecting BGA contact: NANDF_D6 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT9 for Mode: ALT3.

Table A-1283. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_2_SELECT_INPUT

Offset	0x0a04 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_2_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1284. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_2_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_2 0: Selecting BGA contact: NANDF_D5 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT10 for Mode: ALT3.

Table A-1285. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_3_SELECT_INPUT

Offset	0x0a08 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_3_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1286. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_3_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_3 0: Selecting BGA contact: NANDF_D4 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT11 for Mode: ALT3.

Table A-1287. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_4_SELECT_INPUT

Offset	0x0a0c (IOMUXC_USBOH3_IPP_IND_UH3_DATA_4_SELECT_INPUT)	Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1288. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_4_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_4 0: Selecting BGA contact: NANDF_D3 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT12 for Mode: ALT3.

Table A-1289. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_5_SELECT_INPUT

Offset: 0x0a10 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_5_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1290. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_5_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_5 0: Selecting BGA contact: NANDF_D2 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT13 for Mode: ALT3.

Table A-1291. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_6_SELECT_INPUT

Offset	0x0a14 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_6_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1292. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_6_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_6 0: Selecting BGA contact: NANDF_D1 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT14 for Mode: ALT3.

Table A-1293. Register: IOMUXC_USBOH3_IPP_IND_UH3_DATA_7_SELECT_INPUT

Offset	0x0a18 (IOMUXC_USBOH3_IPP_IND_UH3_DATA_7_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1294. Register IOMUXC_USBOH3_IPP_IND_UH3_DATA_7_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_data_7 0: Selecting BGA contact: NANDF_D0 for Mode: ALT5. 1: Selecting BGA contact: DISP2_DAT15 for Mode: ALT3.

Table A-1295. Register: IOMUXC_USBOH3_IPP_IND_UH3_DIR_SELECT_INPUT

Offset: 0x0a1c (IOMUXC_USBOH3_IPP_IND_UH3_DIR_SELECT_INPUT) — Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1296. Register IOMUXC_USBOH3_IPP_IND_UH3_DIR_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_dir 0: Selecting BGA contact: NANDF_CS5 for Mode: ALT7. 1: Selecting BGA contact: DISP2_DAT1 for Mode: ALT3.

Table A-1297. Register: IOMUXC_USBOH3_IPP_IND_UH3_NXT_SELECT_INPUT

Offset	0x0a20 (IOMUXC_USBOH3_IPP_IND_UH3_NXT_SELECT_INPUT)															Access: User read / write
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1298. Register IOMUXC_USBOH3_IPP_IND_UH3_NXT_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_nxt 0: Selecting BGA contact: NANDF_RB2 for Mode: ALT6. 1: Selecting BGA contact: DISP2_DAT7 for Mode: ALT3.

Table A-1299. Register: IOMUXC_USBOH3_IPP_IND_UH3_STP_SELECT_INPUT

Offset: 0x0a24 (IOMUXC_USBOH3_IPP_IND_UH3_STP_SELECT_INPUT) Access: User read / write

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table A-1300. Register IOMUXC_USBOH3_IPP_IND_UH3_STP_SELECT_INPUT Bits Description

Field	Description
31-1	Reserved
0 DAISY	Selecting Pads Involved in Daisy Chain. Instance: usboh3, In Pin: ipp_ind_uh3_stp 0: Selecting BGA contact: NANDF_CS4 for Mode: ALT7. 1: Selecting BGA contact: DISP2_DAT6 for Mode: ALT3.