Freescale Semiconductor
Application Note

Document Number: AN4202
Rev. 0, 2/2011

i.MX51–Modifying WinCE 6.0™ Bootloader and Kernel for Alternate SDRAM Support

by Multimedia Applications Division
Freescale Semiconductor, Inc.
Austin, TX

Software supplied in the i.MX51 application processor Windows Embedded CE 6.0^{TM} (WinCE) board support package (BSP) is configured to support the synchronous dynamic random access memory (SDRAM) specified in the i.MX51 evaluation kit (EVK) reference design. This application note describes changes that may be required to adapt the bootloader and kernel software supplied in the BSP to support a different SDRAM. This document focuses on the basic SDRAM parameters, and SDRAM configuration options of the i.MX51application processor and highlights files within the BSP that may require modification.

A spreadsheet file, `ESDRAMC_Config_Tool.xls`, is provided along with this application note. This spreadsheet provides information required to configure the enhanced SDRAM controller (ESDRAMC) registers with the appropriate memory timing parameters. Note that the i.MX51 Reference manual uses a number of different abbreviations to refer to the enhanced synchronous DRAM Controller. These are: eSDCTL, ESDCTL, and ESDRAMC. This document will use ESDRAMC exclusively.

Contents

1 Introduction

SDRAM selection is a fundamental aspect of any design that uses the i.MX51 because the processor is used in a wide range of applications. The SDRAM employed in a specific product design may differ from the one used in the evaluation kit (EVK) design supplied with the WinCE BSP. The following sections provide information that is useful in making any necessary changes to the BSP software when modification to the configuration of the integrated enhanced synchronous dynamic random assess memory controller (ESDCTL) is required.

2 Enhanced Synchronous Dynamic Random Access memory controller (ESDRAMC)

This section gives an overview of the ESDRAMC and describes its various configuration options.

2.1 Overview

The ESDRAMC consists of the following components:

- SDRAM command state-machine controller
- Bank register (page and bank-address comparators)
- Row/column address multiplexer
- Configuration registers
- Refresh request counter
- Command sequencer
- Size logic (splitting access)
- Data path (data aligner/multiplexer)
- Low power double data rate (LPDDR) interface
- Power down timer

The ESDRAMC features are as follows:

- Supports dynamic frequency and voltage changes in the mobile double data rate (MDDR) and double data rate 2 (DDR2) modes:
  - Negotiation for the frequency changes in the system
  - Constant delay line measurement to allow the voltage changes
- Optimizes consecutive memory accesses through memory command anticipation (latency hiding):
  - Hides latency by optimizing the commands to both the chip selects (command anticipation)
  - Preparation of valid waiting AMBA 3 advanced extensible Interface (AXI) access in terms of ACTIVE and PRECHARGE commands while serving other accesses
  - Preparation of predicted next miss access in terms of ACTIVE and PRECHARGE commands while serving other accesses
  - Keeps track of the open-memory pages
  - Bank wise memory-address mapping

- — MDDR burst-length configuration of 8
- — DDR2 burst-length configuration of 4
- — AXI bus compliant
- — Shared address and command bus to the memory devices

- Supports LPDDR or DDR2 devices of four or eight banks with memory capacity of 64 Mbytes, 128 Mbytes, 256 Mbytes, 512 Mbytes, 1 Gbyte, or 2 Gbytes:
  - — Two independent chip selects
  - — Up to 256 Mbytes per chip select
  - — Activates up to eight banks simultaneously per chip select for the DDR2 and MDDR devices
  - — Supports only column address strobe 3 (CAS 3) devices
  - — Joint Electron Device Engineering Council® (JEDEC) standard pinout operation
- Supports ×16/×32 mDDR/DD2 DDR400 devices
- Supports on-die termination (ODT) for the DDR2 devices
- Supports differential and single-ended DQS modes
- Software configurable for different system and memory-device requirements:
  - — ×16/×32 memory data bus width
  - — Number of row and column addresses
  - — Row cycle delay (t_{RC})
  - — Auto-refresh cycle delay (t_{RFC})
  - — Row precharge delay (t_{RP})
  - — Row to column delay (t_{RCD})
  - — Load mode register (LMR) to ACTIVE command (t_{MRD})
  - — WRITE to PRECHARGE command (t_{WR})
  - — WRITE to READ command for LPDDR memories (t_{WTR})
  - — LPDDR exit power down to the immediate valid command delay (t_{XP})
  - — ACTIVE to PRECHARGE command (t_{RAS})
  - — ACTIVE to ACTIVE command (t_{RRD})
  - — Exit self-refresh to any command (t_{XSR}/t_{XSRD} /t_{SRFX})
- Built-in auto refresh timer and state machine
- Hardware and software supported self-refresh entry and exit:
  - — Data valid during the system reset and low-power modes
  - — Auto power down timer (one per chip select)
  - — Automatic self refresh timer (one per chip select)
  - — Auto precharge timer (one per bank for each chip select)
- Supports deep power-down entry and exit in the LPDDR software
- Supports AXI bursts of up to 8:
  - — Supports bus transfers of 32 and 64 bits

- — Supports 8 and 16 bit transfers in single access
- — Fixed burst type is not supported (except for a burst length of 1)
- Supports page boundary crossing of up to 4 Kbytes in the AXI address:
  Generates the `PRECHARGE` and `ACTIVE` commands in the immediate row when page boundary on the AXI bus is crossed

2.2 Configuration Options

The ESDRAMC has fifteen configuration registers that are used to configure the features of both chip selects. However, this discussion focuses only on five of these configuration registers. These are the Enhanced SDRAM Control Register 0 (ESDCTL0), the Enhanced SDRAM Control Register 1 (ESDCTL1), the ESDCTL Configuration Register 0 (ESDCFG0), the ESDCTL Configuration Register 1(ESDCFG1) and the ESDCTL Miscellaneous Register (ESDMISC). The spreadsheet file, `ESDRAMC_Config_Tool.xls`, which is provided along with this application note calculates the specific values that must be written to the ESDCTL configuration (ESDCFG0/ESDCFG1) and control registers (ESEDCTL0/ESDCTL1) based on input parameters you supply. See *MCIMX51 Multimedia Applications Processor Reference Manual* (MCIMX51RM) for more detailed information on these registers.

The five ESDRAMC configuration registers to be considered in this document are described as follows:

- ESDCTL0-1. The i.MX51 processor has two ESDCTLx registers, one for each chip select. These registers control various memory and control settings. Table 1 gives a description of the ESDCTLx register fields.

Table 1. ESDCTLx Register Fields

Field	Description
SDE	ESDRAMC enable This control bit enables/disables the ESDRAMC. The reset value of this bit is “0”. When disabled, no clocks or clock enable signals will be sent to memory. Writing one to this bit in both ESDCTL0 and ESDCTL1enables the ESDRAMC for both chip selects. The transition from 0 to 1 triggers an SDRAM initialization sequence. Initialization requires 400 microseconds. During this time the DRAM is not accessible. Clearing those bits disables the ESDRAMC. By disabling both bits (SDE0 and SDE1) all clocks within the module are gated. All internal registers are still accessible.
SREFR	SDRAM refresh rate This 3 bit field enables/disables SDRAM refresh cycles and controls the refresh rate. Refresh cycles are referenced to a 32 kHz clock. At each falling edge 1, 2, 4, 8 or 16 rows are refreshed as determined by this bit field. Multiple refresh cycles will be separated by the row cycle delay specified in the SRC control field. Refresh is disabled by hardware reset.
ROW	Row address width This control field specifies the number of row addresses used by the memory array. The value in this field affects the way an incoming address from the AXI bus is decoded.
DBL_tRFC	Double t_{RFC} value This bit must be set when the Auto-refresh to command delay exceeds 25 clock cycles. See the description for t_{RFC} in Table 2 for more information.
COL	Column address width This field specifies the number of column address bits in the memory array and determines how an incoming address is decoded.

Table 1. ESDCTLx Register Fields (continued)

Field	Description
DSIZ	SDRAM memory data width This field defines the width of the SDRAM memory and its alignment on the external data bus. 16-bit ports may be aligned to either the high or low half word to equalize capacitive loading on the bus. Data qualifier mask control outputs must be matched to the selected data bus alignment. Memories aligned to D[31:16] use DQM2 and DQM3. Memories aligned to D[15:0] use DQM0 and DQM1.
SRT	Self refresh timer This field determines whether the SDRAM will be placed in a Self Refresh condition after a selectable delay from the last access. The Self Refresh time-out can be triggered on either the absence of an active bank (SRT=01) or a clock (ESDCTL_CLK) count from the last access (SRT=10 or 11). Count based time-outs do not force the SDRAM into an idle condition (for example, any active banks remain open). The Self Refresh timer feature is disabled by hardware reset.
PWDT	Power down timer This field determines whether the SDRAM will be placed in a Power Down condition after a selectable delay from the last access. The Power Down time-out can be triggered on either the absence of an active bank (PWDT=01) or a clock (ESDCTL_CLK) count from the last access (PWDT=10 or 11). Count based time-outs sends a `Precharge` all to the SDRAM. The Power Down timer feature is disabled by hardware reset.

- ESDCFG0-1. Table 2 gives a description of the ESDRAMC Timing Parameters controlled by the ESDCFGx registers.

NOTE

Timing parameters configured via fields within the following registers are expressed in integer numbers of bus clock cycles. To convert this to time, multiply the number of cycles specified by the bus clock period (bus clock period = 1 / bus frequency).

Table 2. ESDRAMC Timing Parameters Configured by ESDCFGx Registers

Symbol	Description
t_{RFC}	`Auto-refresh` to a command delay This is the idle delay after the `auto-refresh` command until another command is issued. In DDR2, this is the idle delay after the `auto-refresh` command before a non-read command is issued. This field encodes 16 delay values from 10 to 25 clock cycles. If the conversion of the SDRAM timing parameter to clock cycles results in a value which is greater than 25, the result is divided by 2, this new value is written using the same encoding, and the DBL_tRFC bit in the ESDCTLx register is set to indicate that the parameter is two times the given value.
t_{XSR}	Exit `self-refresh` to a command delay This control field determines the minimum delay in issuing a valid command to the SDRAM after exiting the self-refresh mode. The t_{XSR} parameter provides the number of clocks that are inserted after exiting the self-refresh mode and a subsequent valid command. Generally, this parameter is listed as t_{XSNR} in the data sheets.
t_{XP}	Exit power down to the immediate valid command delay This control field determines the minimum delay in issuing a valid command to the memory device after exiting the power-down mode. The t_{XP} parameter provides the number of clocks that are inserted after exiting the power-down mode until issuing a subsequent new valid command.
t_{WTR}	WRITE to READ command delay Data for any write burst is followed by a subsequent READ command. This field specifies the minimum delay between the two commands.

Table 2. ESDRAMC Timing Parameters Configured by ESDCFGx Registers (continued)

Symbol	Description
t_{RP}	Row precharge delay This control bit determines the number of idle clocks that must be inserted between the PRECHARGE command and immediate row activate command of the same bank.
t_{MRD}	LOAD MODE REGISTER (LMR) command cycle time These control bits determine the minimum number of idle clocks that are required after an LMR command.
t_{RAS}	ACTIVE to PRECHARGE command delay These control bits determine the minimum number of clocks that are required between the ACTIVE and PRECHARGE commands of the same bank.
t_{RRD}	Bank A active to Bank B active command delay A subsequent ACTIVE command to a different row in the same bank can be issued only after the previous active row has been closed (precharged). However, a subsequent ACTIVE command to another bank can be issued while the first bank is being accessed, which results in the reduction of total row-access overhead. This field specifies the minimum number of idle clocks that will be inserted between these two commands.
t_{WR}	WRITE to PRECHARGE command delay Data for a fixed length write burst is generally followed by or truncated with the PRECHARGE command to the same bank (provided that the auto precharge is not activated). This field specifies the minimum delay the controller will insert between these two commands.
t_{RCD}	SDRAM row to column delay This field determines the number of clocks that are inserted between a row ACTIVATE command and the subsequent READ or WRITE command of the same bank.
t_{RC}	ACTIVE to ACTIVE command delay in the same bank.

- Enhanced SDRAM miscellaneous register (ESDMISC)—configures various memory and control settings for the ESDRAMC. Table 3 gives a description for the ESDMISC register fields.

Table 3. ESDMISC Field Definitions

Field	Description
CS0_RDY	External device status on CS0 This read-only status bit indicates the status of the external memory devices. This bit is cleared at reset or during deep power-down entry of the DRAM. This bit is set 400 µs after the external memory wake-up period.
CS1_RDY	External device status on CS1 This read-only status bit indicates the status of the external memory devices. This bit is cleared at reset or during deep power down entry of the DRAM. This bit is set after 400 µs of the external memory wake-up period.
ODT_IDLE_ON	On-Die Termination (ODT) behavior in the IDLE mode.
SDCLK_EXT	SDCLK extension In cases that require shutting off SDCLK, setting this bit will keep SDCLK toggling for two extra clocks after CKE goes low.
TERM_CTL3	This 2-bit field determines the termination resistance of the DQS[3] and DQ[31–24] pads when DDR2, which supports ODT mode, performs a read operation.
TERM_CTL2	This 2-bit field determines the termination resistance of the DQS[3] and DQ[23–16] pads when DDR2, which works in the ODT mode, performs a read operation.

Table 3. ESDMISC Field Definitions (continued)

Field	Description
TERM_CTL1	This 2-bit field determines the termination resistance of the DQS[1] and DQ[15–8] pads when DDR2, which works in the ODT mode, performs a read operation.
TERM_CTL0	This 2-bit field determines the termination resistance of the DQS[0] and DQ[7–0] pads when DDR2, which works in the ODT mode, performs a read operation.
AP_BIT	Auto-precharge bit location SDRAM devices utilize an address bit to modify the meaning of a PRECHARGE command. When this bit is set, SDRAM device treats a PRECHARGE command as a PRECHARGE ALL command. The JEDEC standard defines this field to be A10, therefore the default value of this field is 10. If the SDRAM device is not JEDEC compliant, then this field must be set to match the device specification.
DIFF_DQS_EN	Differential DQS-mode enable Enables differential DQS mode when working with a DDR2 device.
AUTO_DLL_PAUSE	Auto-pause enable Enables auto pause of 200 cycles between issuing the delay locked loop (DLL) reset during the device startup (for DDR2 devices) and the first READ command.
ODT_EN	ODT enable Enables the use of ODT control output bits to the memory for DDR2 devices.
BI_ON	Bank interleaving bit This bit indicates whether the system is interleaving the address or not.
FRC_MSR	Force measurement When this bit is set, the measurement unit starts a new measurement until this bit is cleared.
MIF3_MODE	Controls the MIF3-mechanism mode.
RALAT	Read additional latency This field determines when the controller retrieves data from the internal FIFO of the ESDCTL register. Using this field, the board/chip delays can be compensated in both low and high frequencies.
DDR2_8_BLANK	DDR2 device with the eight banks in use This bit is common for both the chip selects.
LHD	Latency hiding disable for read operations When this bit is set, the immediate read access is allowed only after the last data of the first read operation is sent for arbitration.
DDR2_EN	Regular (non mobile) DDR2 device enable This bit is common for both the chip selects.
DDR_EN	Regular (non mobile) DDR1 device enable This bit is common for both the chip selects.
RST	Reset This bit generates a local reset for the ESDRAMC. The ESDRAMC registers are not affected by this software initiated reset in order to keep the refresh mechanism active as initially configured. Therefore, any data read is not validated. To terminate any active bursts, a BURST TERMINATE command is issued to the memory after the soft reset. This prevents potential contention on the data pads. **Note:** After soft reset, the PRECHARGE ALL command must be issued prior to normal usage of the SDRAM.

NOTE

> See *MCIMX51 Multimedia Applications Processor Reference Manual* (MCIMX51RM) for more information.

3 Modifying BSP Software to Support Alternate SDRAM

The main function of the bootloader is to run the basic initialization code on the board, place the run-time image into the memory, and jump into the OS-startup function. Files within the BSP may require modification when the SDRAM in a specific product design differs from that specified by the EVK design.

The following files affect the way the code configures the ESDRAMC:

- `xldr.s` contains the low-level code that configures the ESDRAMC registers.
- `config.bib` is used by the platform builder to create the OS binary image. It contains two sections, MEMORY and CONFIG. The MEMORY section defines the memory table for the run-time image.
- `image_cfg.h` contains the base address that is used in images. This file is used in the `*.c` files.
- `image_cfg.inc` contains the memory map addresses that are based on images. This file is used only in the low level `*.s` files.
- `Oemaddrtab_cfg.inc` contains the original equipment manufacturer (OEM) address table definition. This table maps the 4-Gbyte physical address space in the 512-Mbyte unmapped space of the kernel.

3.1 Procedure

The configuration tool assumes that you are using a single SDRAM device that uses one or both chip select pins (DRAM_CS0-1) of the i.MX51 or, if you are using multiple devices, that they are all the same memory device part number.

The following is the process that must be followed to modify the Bootloader and Kernel code to support a SDRAM device type other than the Micron part specified in the i.MX51 EVK reference design. Refer to the code excerpts below for this discussion.

1. Determine the following timing values based on your selected SDRAM and enter them into the `ESDCRAM_Config_Tool.xls` spreadsheet.
   a) Find the values t_{RFC}, t_{XSR}, t_{WTR}, t_{RP}, t_{RAS}, t_{RRD}, t_{WR}, t_{RCD}, and t_{RC} (expressed in nanoseconds) from the SDRAM datasheet. Note that the t_{XSR} value is usually listed as t_{XSNR} in SDRAM datasheets.
   b) Find the timing parameters t_{XP} and t_{MRD} (expressed in minimum number of clock periods).
   c) Determine the Row Address Width (in bits), Column Address Width (in bits), and Memory Data Width (16-bit aligned to D[31:16], 16-bit aligned to D[15:0], or 32-bit).
   d) Set Refresh Rate (SREFR) to Disabled.
   e) Determine Self Refresh Timer value (This should be one of the following: Any time no banks are active, 64 clocks, 128 clocks, or Disabled).
   f) Determine Power Down Timer value (This should be one of the following: Any time no banks are active, 256 clocks, 512 clocks, or Disabled).

   g) Chose SDRAMC Enable = Yes.

2. The spreadsheet calculates the correct values for the ESDRAMC Configuration Registers (ESDCFG0 / ESDCFG1) and the ESDRAMC Control Registers (ESDCTL0 / ESDCTL1).
3. Make the following changes to the `ESDCTLSetup` routine in the `xldr.s` file:
   a) Leave the value loaded into R0 at line `<aa>` `=0x00000200` unless you are experiencing SDRAM signal integrity issues on your board. If you are, try changing the value to `=0x00000000`.
   b) Change the value loaded into R0 on line `<ab>` if the ESDRAMC Control Register value from the spreadsheet differs from `=0x82a20000`.
   c) Generally there is no need to change the value written to the ESDMISC register in lines `<ac>` and <`ac+1`>.
   d) Change the value loaded into R0 on line `<ad>` if the ESDRAMC Configuration Register value from the spreadsheet differs from `=0x333574aa.`
   e) Go back to the spreadsheet. Change the SDRAM Refresh Rate [SREFR] to the correct value for your SDRAM. Hit recalculate in Excel to generate a new ESDRAM Control Register value.
   f) Change the value loaded into R0 on line `<ae>` if the ESDRAMC Control Register value from the spreadsheet differs from `=0xb2a20000`.
4. Make changes to `Config.bib` as required:
   a) If your memory size is other than 64, 128, or 256 MB change lines `<ba>` through `<bb>`, to redefine the starting address for the FEC memory region.
   b) In lines `<bc>` through `<bd>` change the definition of RAM_SIZE based on your SDRAM size.
   c) In lines `<be>` and `<bf>` change the value assigned to RAM_SIZE based on your SDRAM's size.
5. Make changes to `image_config.h` as required:
   a) At line `<ca>` change the value assigned to the symbol `IMAGE_BOOT_RAMDEV_RAM_SIZE` to match your SDRAM size.
   b) Change the value assigned to the symbol `IMAGE_SHARE_FEC_RAM_OFFSET` in lines `<db>` through `<dc>.`
6. Make the following changes to `image_cfg.inc` as required:
   a) Change the value assigned to `IMAGE_BOOT_RAMDEV_RAM_SIZE` on line `<da>` if your SDRAM size is other than 128 MB.
   b) Rework `ifdef` statements on lines `<cx>` through `<cy>` to assign a new value to the symbol `IMAGE_SHARE_FEC_RAM_OFFSET.`
7. Make the following changes to `Oemaddrtab_cfg.inc` as required:
   a) In lines `<ea>` through `<eb>` change the RAM image mapping as required for your SDRAM. To do this, you only need to change the memory size number of the entries. The use of one or two entries to map the complete SDRAM will depend on the number of chip selects used to allocate the memory.

3.2 xldr.s

The `xldr.s` file must be modified whenever the AC parameters of the SDRAM change. This file sets the configuration for the IOMUX PADS and ESDCTLx, ESDCFGx, and ESDMISC registries. The `xldr.s` file fragment is as follows:

NOTE

In the code listed below, line numbers have been added, written as **`<xx>`**, to guide the user in making changes to customize SDRAM parameters. Comments have also been added which do not appear in the BSP code.

```
;-------------------------------------------------------------------------------
;
;         Function:  ESDCTLSetup
;
;         Configure DDR memory
;
;   NOTE: It is important for this function to be defined (& loaded last in address map) so that
;         the SD functions are part of initial 2K loaded in by ROM code. That will allow copying

;            remaining 2K from SD/MMC card.
;
;        Parameters:
;         No parameters
;
;        Returns:
;         No return value
;-------------------------------------------------------------------------------
         LEAF_ENTRY ESDCTLSetup

         IF :DEF: BSP_SI_VER_TO2

         ;
         ; Configure IOMUX for DDR2
         ;

         ; IOMUXC_SW_PAD_CTL_GRP_INMODE1          DDR2 mode
<aa>     ldr     r0, =0x00000200      //Change if DDR2 memory is not used
         ldr     r1, =0x73fa88a0
         str     r0, [r1]

         //May Change according to your hardware design
         ; IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT1       Pull-D, HIGH DS      //Do not change
         ldr     r0, =0x000020c5                                       //Do not change
         ldr     r1, =0x73fa850c                                       //Do not change

         str     r0, [r1]                                              //Do not change

         ; IOMUXC_SW_PAD_CTL_PAD_EIM_SDODT0       Pull-D, HIGH DS      //Do not change
         ldr     r0, =0x000020c5                                       //Do not change
         ldr     r1, =0x73fa8510                                       //Do not change
         str     r0, [r1]                                              //Do not change

         ; IOMUXC_SW_PAD_CTL_GRP_DDR_A0           MEDIUM DS            //Do not change
         ldr     r0, =0x00000002                                       //Do not change
         ldr     r1, =0x73fa883c                                       //Do not change
         str     r0, [r1]                                              //Do not change
```

```
; IOMUXC_SW_PAD_CTL_GRP_DDR_A1           MEDIUM DS            //Do not change
ldr     r0, =0x00000002                                       //Do not change
ldr     r1, =0x73fa8848                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK       MAX DS               //Do not change
ldr     r0, =0x000000e7                                       //Do not change
ldr     r1, =0x73fa84b8                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0       Pull/Keep Disable    //Do not change
ldr     r0, =0x00000045                                       //Do not change
ldr     r1, =0x73fa84bc                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1       Pull/Keep Disable    //Do not change
ldr     r0, =0x00000045                                       //Do not change
ldr     r1, =0x73fa84c0                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2       Pull/Keep Disable    //Do not change
ldr     r0, =0x00000045                                       //Do not change
ldr     r1, =0x73fa84c4                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3       Pull/Keep Disable    //Do not change
ldr     r0, =0x00000045                                       //Do not change
ldr     r1, =0x73fa84c8                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_GRP_DDRPKS           PUE = Keeper         //Do not change
ldr     r0, =0x00000000                                       //Do not change
ldr     r1, =0x73fa8820                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS         MEDIUM DS            //Do not change
ldr     r0, =0x00000003                                       //Do not change
ldr     r1, =0x73fa84a4                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS         MEDIUM DS            //Do not change
ldr     r0, =0x00000003                                       //Do not change
ldr     r1, =0x73fa84a8                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE        MEDIUM DS            //Do not change
ldr     r0, =0x000000e3                                       //Do not change
ldr     r1, =0x73fa84ac                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0      MEDIUM DS            //Do not change
ldr     r0, =0x000000e3                                       //Do not change
ldr     r1, =0x73fa84b0                                       //Do not change
str     r0, [r1]                                              //Do not change

; IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1      MEDIUM DS            //Do not change
```

```
        ldr     r0, =0x000000e3                                       //Do not change
        ldr     r1, =0x73fa84b4                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0         MEDIUM DS            //Do not change
        ldr     r0, =0x000000e3                                       //Do not change
        ldr     r1, =0x73fa84cc                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1         MEDIUM DS            //Do not change
        ldr     r0, =0x000000e2                                       //Do not change
        ldr     r1, =0x73fa84d0                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; DRAM_D31-24                            MAX DS               //Do not change
        ldr     r0, =0x00000006                                       //Do not change
        ldr     r1, =0x73fa882c                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; DRAM_D0-7                              MAX DS               //Do not change
        ldr     r0, =0x00000006                                       //Do not change
        ldr     r1, =0x73fa88a4                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; DRAM_D8-15                             MAX DS               //Do not change
        ldr     r0, =0x00000006                                       //Do not change
        ldr     r1, =0x73fa88ac                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ; DRAM_D16-23                            MAX DS               //Do not change
        ldr     r0, =0x00000006                                       //Do not change
        ldr     r1, =0x73fa88b8                                       //Do not change
        str     r0, [r1]                                              //Do not change

        ;
        ; Configure ESDCTL for 32-bit DDR
        ;
        ldr     r1, =(CSP_BASE_REG_PA_ESDCTL)


        ; 13 ROW, 10 COL, 32Bit, SREF=4 Micron Model
        ; CAS=3,  BL=4
<ab>    ldr     r0, =0x82a20000  //Change based on your memory's parameters
        str     r0, [r1, #ESDCTL_ESDCTL0_OFFSET]
        str     r0, [r1, #ESDCTL_ESDCTL1_OFFSET]

<ac>    ldr     r0, =0x000ad0d0
        str     r0, [r1, #ESDCTL_ESDMISC_OFFSET]

        ; tRFC=13 tXSR=28 tXP=2 tRP=3 tMRD=2 tRAS=8 tRRD=2 tWR=3 tRCD=3 tRC=11
<ad>    ldr     r0, =0x333574aa // Change based on your memory's parameters
        str     r0, [r1, #ESDCTL_ESDCFG0_OFFSET]
        str     r0, [r1, #ESDCTL_ESDCFG1_OFFSET]

        ; Init DRAM on CS0
        ldr     r0, =0x04008008
        str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
```

```
         ldr     r0, =0x0000801a
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0000801b
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00448019
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x07328018
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x04008008
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008010
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008010
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x06328018
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x03808019
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00408019
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008000
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]

         ; Init DRAM on CS1
         ldr     r0, =0x0400800c
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0000801e
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0000801f
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0000801d
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0732801c
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0400800c
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008014
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008014
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0632801c
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0380801d
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x0040801d
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
         ldr     r0, =0x00008004
         str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]


         ; 13 ROW, 10 COL, 32Bit, SREF=4 Micron Model
<ae>     ldr     r0, =0xb2a20000  //Change based on your memory’s parameters (activates self
refresh [SREF])
         str     r0, [r1, #ESDCTL_ESDCTL0_OFFSET]
         str     r0, [r1, #ESDCTL_ESDCTL1_OFFSET]

         ; RALAT=01, NO ODT
```

```
ldr     r0, =0x000ad6d0
str     r0, [r1, #ESDCTL_ESDMISC_OFFSET]

ldr     r0, =0x90000000
str     r0, [r1, #ESDCTL_ESDCDLYGD_OFFSET]

; Clear "configuration request" bit
ldr     r0, =0x00000000
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]


ELSE
;
; Configure ESDCTL for 32-bit DDR
;
ldr     r1, =(CSP_BASE_REG_PA_ESDCTL)

; Enable CSD0
ldr     r0, =0x80000000
str     r0, [r1, #ESDCTL_ESDCTL0_OFFSET]

; Precharge
ldr     r0, =0x04008008
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]

; 2 Refresh commands
ldr     r0, =0x00008010
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]

; LMR with CAS=3 and BL=3
ldr     r0, =0x00338018
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]

; 13 ROW, 10 COL, 32Bit, SREF=4 Micron Model
ldr     r0, =0xB2220000
str     r0, [r1, #ESDCTL_ESDCTL0_OFFSET]

; Timing parameters
ldr     r0, =0xB02567A9
str     r0, [r1, #ESDCTL_ESDCFG0_OFFSET]

; MDDR enable, BI_ON=1, RLAT=2
ldr     r0, =0x000a1104
str     r0, [r1, #ESDCTL_ESDMISC_OFFSET]

; Normal mode
ldr     r0, =0x00000000
str     r0, [r1, #ESDCTL_ESDSCR_OFFSET]

ENDIF

RETURN

END
```

3.3 Config.bib

The `Config.bib` file must be modified only when the size of the RAM memory changes. The `Config.bib` file fragment is as follows:

```
;**************************** MEMORY SECTION ***************************
MEMORY

; ----------------- Reserved Region Mapping (2 MB) ---------------------
;   Start Addr      End Addr        Mem Type    Region Name     Size
;   0x80000000      0x80000FFF      SDRAM       ARGS              4 KB
;   0x80001000      0x80008FFF      SDRAM       CSPDDK           32 KB
;   0x80009000      0x801FFFFF      SDRAM       PP             2012 KB
;   0x8FFFC000      0x8FFFFFFF      SDRAM       FEC              16 KB
; ----------------------------------------------------------------------
;                 Name        Address     Size        Type
                  ARGS        80000000    00001000    RESERVED
                  CSPDDK      80001000    00008000    RESERVED
                  PP          80009000    001F7000    RESERVED
; Obsolete region, free them for more Program Memory
;                 VPU         87700000    00080000    RESERVED
;                 IPU         87780000    00880000    RESERVED
//i.MX51EVK BSP already supports 256MB and 128MB and 64MB of RAM, if a different RAM size is
needed it will need to be added
// This section reserves memory for FEC at the end of the memory map
// The following line is designated <ba> as outlined in the procedure of section 3.1.
#if "$(IMGRAM256)" == "1"                                             ;May require change
FEC         8FFFC000    00004000    RESERVED                          ;May require change
#else                                                                 ;May require change
#if "$(IMGRAM128)" == "1"                                             ;May require change
                  FEC         87FFC000    00004000    RESERVED        ;May require change
#else                                                                 ;May require change
                  FEC         9BFFC000    00004000    RESERVED        ;May require change
#endif                                                                 ;May require change
//The previous line is designated <bb> as outlined in the procedure of section 3.1.
#endif

IF IMGFLASH !

        #define NK_START    80200000
IF IMGTINY !

IF IMGSMALLNK !
        #define NK_SIZE     05E00000
        #define RAM_START   86000000
ENDIF
IF IMGSMALLNK
        #define NK_SIZE     02F00000
        #define RAM_START   83100000
ENDIF

ENDIF
IF IMGTINY
        #define NK_SIZE     00500000
        #define RAM_START   80580000
ENDIF
```

```
; --------------------------- RAM image (Windows CE) -------------------
;   Start Addr      End Addr        Mem Type    Region Name     Size
;   0x80000000      0x801FFFFF      SDRAM       reserved         2 MB
;   0x80200000      0x85FFFFFF      SDRAM       NK              94 MB
;   0x86000000      0x9BFFBFFF      SDRAM       RAM            352 MB - 16KB
;   0x9BFFC000      0x9BFFFFFF      SDRAM       reserved        16 KB
; ---------------------------------------------------------------------
Following line is designated <bc> as outlined in the procedure of section 3.1.
#if "$(IMGRAM256)" == "1"                                     ;May require change
IF IMGSMALLNK !                                               ;May require change
        #define RAM_SIZE     09FFC000                         ;May require change
ENDIF                                                         ;May require change
IF IMGSMALLNK                                                 ;May require change
        #define RAM_SIZE     0CEFC000                         ;May require change
ENDIF                                                         ;May require change
#else                                                         ;May require change
#if "$(IMGRAM128)" == "1"                                     ;May require change
IF IMGSMALLNK !                                               ;May require change
        #define RAM_SIZE     01FFC000                         ;May require change
ENDIF                                                         ;May require change
IF IMGSMALLNK                                                 ;May require change
        #define RAM_SIZE     04EFC000                         ;May require change
ENDIF                                                         ;May require change
#else                                                         ;May require change
IF IMGSMALLNK !                                               ;May require change
        #define RAM_SIZE     15FFC000                         ;May require change
ENDIF                                                         ;May require change
IF IMGSMALLNK                                                 ;May require change
        #define RAM_SIZE     18EFC000                         ;May require change
ENDIF                                                         ;May require change
#endif                                                        ;May require change
#endif                                                        ;May require change
ENDIF                                                         ;May require change
The previous line is designated <bd> as outlined in the procedure of section 3.1.
IF IMGFLASH
        IF IMGEBOOT
                #define NK_START    90040000
                #define NK_SIZE     03FC0000
                #define RAM_START   80200000
; --------------- FLASH image with EBOOT (Windows CE) ------------------
;   Start Addr      End Addr        Mem Type    Region Name     Size
;   0x90000000      0x9007FFFF      FLASH       resident EBOOT   512 KB
;   0x90080000      0x93FFFFFF      FLASH       NK             63.50 MB
;   0x80000000      0x801FFFFF      SDRAM       reserved            2 MB
;   0x80200000      0x876FFFFF      SDRAM       RAM               126 MB

----------------------------------------------------------------------
<be>            #define RAM_SIZE     07E00000                 ;May require change
        ENDIF

        IF IMGEBOOT !
                #define NK_START    90000000
                #define NK_SIZE     04000000
                #define RAM_START   80200000
; ------------ FLASH image without EBOOT (Windows CE) ------------------
;   Start Addr      End Addr        Mem Type    Region Name     Size
;   0x90000000      0x83FFFFFF      FLASH       NK              64 MB
;   0x80000000      0x801FFFFF      SDRAM       reserved         2 MB
;   0x80200000      0x876FFFFF      SDRAM       RAM            126 MB
; ----------------------------------------------------------------------
<bf>             #define RAM_SIZE     07E00000               ;May require change
        ENDIF
ENDIF

; ----------------------------------------------------------------------
;                   Name        Address     Size        Type
NK          $(NK_START)  $(NK_SIZE)  RAMIMAGE
RAM         $(RAM_START) $(RAM_SIZE) RAM


;**************************** CONFIG SECTION ***************************
```

3.4 Image_config.h

The `Image_config.h` file must be modified only when the size of the SDRAM changes. Only the section that sets the total available RAM and fast ethernet controller (FEC) memory buffer location must be changed. The `Image_config.h` file fragment is as follows:

```
//------------------------------------------------------------------------------
// RAM image defines
#define IMAGE_BOOT_RAMDEV_RAM_PA_START      CSP_BASE_MEM_PA_CSD0  // 128MB RAM CSD0
The following line is designated <ca> as outlined in the procedure of section 3.1.
#define IMAGE_BOOT_RAMDEV_RAM_SIZE          (128*1024*1024)   //May require change
#define IMAGE_BOOT_RAMDEV_RAM_PA_END
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_RAMDEV_RAM_SIZE-1)

#define IMAGE_SHARE_ARGS_RAM_OFFSET         0
#define IMAGE_SHARE_ARGS_RAM_PA_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_SHARE_ARGS_RAM_OFFSET)
#define IMAGE_SHARE_ARGS_UA_START           OALPAtoVA(IMAGE_SHARE_ARGS_RAM_PA_START, FALSE)
#define IMAGE_SHARE_ARGS_RAM_SIZE           (4*1024)

#define IMAGE_WINCE_CSPDDK_RAM_OFFSET
(IMAGE_SHARE_ARGS_RAM_OFFSET+IMAGE_SHARE_ARGS_RAM_SIZE)
#define IMAGE_WINCE_CSPDDK_RAM_PA_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_WINCE_CSPDDK_RAM_OFFSET)
#define IMAGE_WINCE_CSPDDK_RAM_SIZE         (32*1024)
#define IMAGE_WINCE_DDKSDMA_RAM_PA_START    (IMAGE_WINCE_CSPDDK_RAM_PA_START)
#define IMAGE_WINCE_DDKSDMA_RAM_SIZE        (16*1024)
#define IMAGE_WINCE_DDKCLK_RAM_PA_START
(IMAGE_WINCE_CSPDDK_RAM_PA_START+IMAGE_WINCE_DDKSDMA_RAM_SIZE)
#define IMAGE_WINCE_DDKCLK_RAM_UA_START     OALPAtoVA(IMAGE_WINCE_DDKCLK_RAM_PA_START, FALSE)
#define IMAGE_WINCE_DDKCLK_RAM_SIZE         (16*1024)

#define IMAGE_BOOT_XLDR_RAM_OFFSET
(IMAGE_SHARE_ARGS_RAM_OFFSET+IMAGE_SHARE_ARGS_RAM_SIZE)
#define IMAGE_BOOT_XLDR_RAM_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_XLDR_RAM_OFFSET)
#define IMAGE_BOOT_XLDR_RAM_SIZE            (4*1024)

#define IMAGE_BOOT_STACK_RAM_OFFSET
(IMAGE_BOOT_XLDR_RAM_OFFSET+IMAGE_BOOT_XLDR_RAM_SIZE)
#define IMAGE_BOOT_STACK_RAM_START
```

```
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_STACK_RAM_OFFSET)
#define IMAGE_BOOT_STACK_RAM_SIZE           (232*1024)

#define IMAGE_BOOT_BOOTPT_RAM_OFFSET
(IMAGE_BOOT_STACK_RAM_OFFSET+IMAGE_BOOT_STACK_RAM_SIZE)
#define IMAGE_BOOT_BOOTPT_RAM_PA_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_BOOTPT_RAM_OFFSET)
#define IMAGE_BOOT_BOOTPT_RAM_SIZE          (16*1024)
#define IMAGE_BOOT_BOOTPT_RAM_PA_END
(IMAGE_BOOT_BOOTPT_RAM_PA_START+IMAGE_BOOT_BOOTPT_RAM_SIZE-1)

#define IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET
(IMAGE_BOOT_BOOTPT_RAM_OFFSET+IMAGE_BOOT_BOOTPT_RAM_SIZE)
#define IMAGE_BOOT_BOOTIMAGE_RAM_PA_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET)
#define IMAGE_BOOT_BOOTIMAGE_RAM_SIZE       (512*1024)
#define IMAGE_BOOT_BOOTIMAGE_RAM_PA_END
(IMAGE_BOOT_BOOTIMAGE_RAM_PA_START+IMAGE_BOOT_BOOTIMAGE_RAM_SIZE-1)

#define IMAGE_BOOT_NANDCACHE_RAM_OFFSET     IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET
#define IMAGE_BOOT_NANDCACHE_RAM_START      IMAGE_BOOT_BOOTIMAGE_RAM_PA_START
#define IMAGE_BOOT_NANDCACHE_RAM_SIZE       IMAGE_BOOT_BOOTIMAGE_RAM_SIZE

#define IMAGE_BOOT_IPLIMAGE_RAM_OFFSET
(IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET+IMAGE_BOOT_BOOTIMAGE_RAM_SIZE)
#define IMAGE_BOOT_IPLIMAGE_RAM_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_IPLIMAGE_RAM_OFFSET)
#define IMAGE_BOOT_IPLIMAGE_SIZE            (256*1024)

#define IMAGE_BOOT_PICTURE_RAM_OFFSET      (0x10000000)
#define IMAGE_BOOT_PICTURE_RAM_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_PICTURE_RAM_OFFSET)
#define IMAGE_BOOT_PICTURE_SIZE            (2*1024*1024)

#define IMAGE_BOOT_NKIMAGE_RAM_OFFSET       (0x200000)
#define IMAGE_BOOT_NKIMAGE_RAM_PA_START
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_NKIMAGE_RAM_OFFSET)
#define IMAGE_BOOT_NKIMAGE_RAM_SIZE
(IMAGE_BOOT_RAMDEV_RAM_SIZE-IMAGE_BOOT_NKIMAGE_RAM_OFFSET)
#define IMAGE_BOOT_NKIMAGE_RAM_PA_END
(IMAGE_BOOT_NKIMAGE_RAM_PA_START+IMAGE_BOOT_NKIMAGE_RAM_SIZE-1)

// PP internal working buffer, code buffer
#define IMAGE_WINCE_PP_RAM_OFFSET          (0x9000)
#define IMAGE_WINCE_PP_RAM_PA_START        (CSP_BASE_MEM_PA_CSD0+IMAGE_WINCE_PP_RAM_OFFSET)
#define IMAGE_WINCE_PP_RAM_SIZE            (1.6*1024*1024)

// FEC DMA buffer
The following line is designated <cb> as outlined in the procedure of section 3.1.
#ifdef IMGRAM256
#define IMAGE_SHARE_FEC_RAM_OFFSET           (0xFFFC000)      //May require change
#else
#ifdef IMGRAM128
#define IMAGE_SHARE_FEC_RAM_OFFSET           (0x7FFC000)      //May require change
#else
#define IMAGE_SHARE_FEC_RAM_OFFSET           (0x1BFFC000)     //May require change
#endif
```

```
The previous line is designated <cc> as outlined in the procedure of section 3.1.
#endif
#define IMAGE_SHARE_FEC_RAM_PA_START             (CSP_BASE_MEM_PA_CSD0+IMAGE_SHARE_FEC_RAM_OFFSET)
#define IMAGE_SHARE_FEC_RAM_SIZE                 (16*1024)


//Reserved video memory 64M
#ifdef IMGRAM256
#else
#ifdef IMGRAM128
#else
#define IMAGE_WINCE_VIDEO_RAM_OFFSET             (0x1C000000)
#define IMAGE_WINCE_VIDEO_RAM_PA_START
(CSP_BASE_MEM_PA_CSD0+IMAGE_WINCE_VIDEO_RAM_OFFSET)
#define IMAGE_WINCE_VIDEO_RAM_SIZE               (64*1024*1024)
#endif
#endif

//------------------------------------------------------------------------------
```

3.5 image_cfg.inc

The image_cfg.inc file must be changed only when the size of the SDRAM changes. Only the section that sets the total available RAM and FEC buffer location must be changed. The image_cfg.inc file fragment is as follows:

```
;;------------------------------------------------------------------------------
;; RAM image defines
IMAGE_BOOT_RAMDEV_RAM_PA_START      EQU     CSP_BASE_MEM_PA_CSD0  ;; 128MB RAM CSD0
IMAGE_BOOT_RAMDEV_RAM_UA_START      EQU     (0xA0000000)
IMAGE_BOOT_RAMDEV_RAM_CA_START      EQU     (0x80000000)
The following line is designated <da> as outlined in the procedure of section 3.1.
IMAGE_BOOT_RAMDEV_RAM_SIZE          EQU     (128*1024*1024)   ;;May require change
IMAGE_BOOT_RAMDEV_RAM_PA_END        EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_RAMDEV_RAM_SIZE-1)

IMAGE_SHARE_ARGS_RAM_OFFSET         EQU     (0)
IMAGE_SHARE_ARGS_RAM_PA_START       EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_SHARE_ARGS_RAM_OFFSET)
IMAGE_SHARE_ARGS_RAM_UA_START       EQU
(IMAGE_BOOT_RAMDEV_RAM_UA_START+IMAGE_SHARE_ARGS_RAM_OFFSET)
IMAGE_SHARE_ARGS_RAM_SIZE           EQU     (4*1024)

IMAGE_BOOT_XLDR_RAM_OFFSET          EQU
(IMAGE_SHARE_ARGS_RAM_OFFSET+IMAGE_SHARE_ARGS_RAM_SIZE)
IMAGE_BOOT_XLDR_RAM_START           EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_XLDR_RAM_OFFSET)
IMAGE_BOOT_XLDR_RAM_SIZE            EQU     (4*1024)

IMAGE_BOOT_STACK_RAM_OFFSET         EQU
(IMAGE_BOOT_XLDR_RAM_OFFSET+IMAGE_BOOT_XLDR_RAM_SIZE)
IMAGE_BOOT_STACK_RAM_SIZE           EQU     (232*1024)
;; Stack grows downward.  Start must be at top of reserved region.
IMAGE_BOOT_STACK_RAM_PA_START       EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_STACK_RAM_OFFSET+IMAGE_BOOT_STACK_RAM_SIZE)
IMAGE_BOOT_STACK_RAM_CA_START       EQU
```

```
(IMAGE_BOOT_RAMDEV_RAM_CA_START+IMAGE_BOOT_STACK_RAM_OFFSET+IMAGE_BOOT_STACK_RAM_SIZE)

IMAGE_BOOT_BOOTPT_RAM_OFFSET        EQU
(IMAGE_BOOT_STACK_RAM_OFFSET+IMAGE_BOOT_STACK_RAM_SIZE)
IMAGE_BOOT_BOOTPT_RAM_PA_START      EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_BOOTPT_RAM_OFFSET)
IMAGE_BOOT_BOOTPT_RAM_SIZE          EQU     (16*1024)
IMAGE_BOOT_BOOTPT_RAM_PA_END        EQU
(IMAGE_BOOT_BOOTPT_RAM_PA_START+IMAGE_BOOT_BOOTPT_RAM_SIZE-1)

IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET     EQU
(IMAGE_BOOT_BOOTPT_RAM_OFFSET+IMAGE_BOOT_BOOTPT_RAM_SIZE)
IMAGE_BOOT_BOOTIMAGE_RAM_PA_START   EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET)
IMAGE_BOOT_BOOTIMAGE_RAM_SIZE       EQU     (512*1024)
IMAGE_BOOT_BOOTIMAGE_RAM_PA_END     EQU
(IMAGE_BOOT_BOOTIMAGE_RAM_PA_START+IMAGE_BOOT_BOOTIMAGE_RAM_SIZE-1)

IMAGE_BOOT_NANDCACHE_RAM_OFFSET     EQU     IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET
IMAGE_BOOT_NANDCACHE_RAM_START      EQU     IMAGE_BOOT_BOOTIMAGE_RAM_PA_START
IMAGE_BOOT_NANDCACHE_RAM_SIZE       EQU     IMAGE_BOOT_BOOTIMAGE_RAM_SIZE

IMAGE_BOOT_IPLIMAGE_RAM_OFFSET      EQU
(IMAGE_BOOT_BOOTIMAGE_RAM_OFFSET+IMAGE_BOOT_BOOTIMAGE_RAM_SIZE)
IMAGE_BOOT_IPLIMAGE_RAM_START       EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_IPLIMAGE_RAM_OFFSET)
IMAGE_BOOT_IPLIMAGE_SIZE            EQU     (256*1024)

IMAGE_BOOT_PICTURE_RAM_OFFSET       EQU     (0x10000000)
IMAGE_BOOT_PICTURE_RAM_START        EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_PICTURE_RAM_OFFSET)
IMAGE_BOOT_PICTURE_SIZE             EQU     (2*1024*1024)

IMAGE_BOOT_NKIMAGE_RAM_OFFSET       EQU     (0x200000)
IMAGE_BOOT_NKIMAGE_RAM_PA_START     EQU
(IMAGE_BOOT_RAMDEV_RAM_PA_START+IMAGE_BOOT_NKIMAGE_RAM_OFFSET)
IMAGE_BOOT_NKIMAGE_RAM_SIZE         EQU
(IMAGE_BOOT_RAMDEV_RAM_SIZE-IMAGE_BOOT_NKIMAGE_RAM_OFFSET)
IMAGE_BOOT_NKIMAGE_RAM_PA_END       EQU
(IMAGE_BOOT_NKIMAGE_RAM_PA_START+IMAGE_BOOT_NKIMAGE_RAM_SIZE-1)

;; PP internal working buffer, code buffer
IMAGE_WINCE_PP_RAM_OFFSET          EQU      (0x9000)
IMAGE_WINCE_PP_RAM_PA_START        EQU      (CSP_BASE_MEM_PA_CSD0+IMAGE_WINCE_PP_RAM_OFFSET)
IMAGE_WINCE_PP_RAM_SIZE            EQU      (1.6*1024*1024)

;; FEC DMA buffer
<db>     IF :DEF: IMGRAM256
IMAGE_SHARE_FEC_RAM_OFFSET          EQU      (0xFFFC000)      ;;May require change
         ELSE
         IF :DEF: IMGRAM128
IMAGE_SHARE_FEC_RAM_OFFSET          EQU      (0x7FFC000)      ;;May require change
         ELSE
IMAGE_SHARE_FEC_RAM_OFFSET          EQU      (0x1BFFC000)     ;;May require change
         ENDIF
<dc>     ENDIF
IMAGE_SHARE_FEC_RAM_PA_START        EQU      (CSP_BASE_MEM_PA_CSD0+IMAGE_SHARE_FEC_RAM_OFFSET)
```

```
IMAGE_SHARE_FEC_RAM_SIZE            EQU      (16*1024)

;Reserved video memory 64M
         IF :DEF: IMGRAM256
         ELSE
         IF :DEF: IMGRAM128
         ELSE
IMAGE_WINCE_VIDEO_RAM_OFFSET        EQU      (0x1C000000)
IMAGE_WINCE_VIDEO_RAM_PA_START     EQU     (CSP_BASE_MEM_PA_CSD0+IMAGE_WINCE_VIDEO_RAM_OFFSET)
IMAGE_WINCE_VIDEO_RAM_SIZE          EQU      (64*1024*1024)
         ENDIF
         ENDIF
```

3.6 Oemaddrtab_cfg.inc

The `Oemaddrtab_cfg.inc` file must be modified only when the size of the SDRAM changes. Only the section that sets the mapping of the RAM memory space must be changed. The `Oemaddrtab_cfg.inc` file fragment is as follows:

```
;------------------------------------------------------------------------------
;
; TABLE FORMAT
;       cached address, physical address, size
;------------------------------------------------------------------------------

g_oalAddressTable

<ea>     IF :DEF: IMGRAM256
         DCD 0x80000000, CSP_BASE_MEM_PA_CSD0, 256           ; May require change
         ELSE
         IF :DEF: IMGRAM128
         DCD 0x80000000, CSP_BASE_MEM_PA_CSD0, 128           ; May require change
         ELSE
         DCD 0x80000000, CSP_BASE_MEM_PA_CSD0, 256           ; May require change
         DCD 0x90000000, CSP_BASE_MEM_PA_CSD1, 192         ; Remaining 64M are reserved for IPU
         ENDIF
<eb>     ENDIF
```

4 Revision History

Table 4 provides a revision history for this application note.

Table 4. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	09/2010	Initial release.

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

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

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

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

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

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.