Freescale Semiconductor
Application Note

Document Number: AN4466
Rev. 1, 05/2013

i.MX53 DDR Calibration

1 Introduction

The purpose of this document is to describe how to perform various calibration processes on the i.MX53 Enhanced DRAM Controller (ESDCTL) for use with DDR3 and LPDDR2 memories. These calibration processes are needed to fine-tune various delay-line parameters in the ESDCTL for optimal performance and functionality at the target frequency of 400 MHz.

2 DDR Calibration Modes

i.MX53 DDR interface supports the following nine calibration processes:

- ZQ calibration—Change the values of on-chip pull-up and pull-down resistors connected to the VCC/2 pins. Unlike all other types of calibration which take place only at the i.MX53 DDR port, ZQ calibration takes place on the DDR device side as well.
- Write leveling calibration—In DDR3 mode, Set DQS, DQ and DM signals of each data byte to their common timing delay relative to the DDR CLK, ADDR and Controls. This calibration is associated with the DDR3 "fly-by" board topology, as described by the JESD79-3E standard.

Contents

- DQS gating calibration—Optimize the timing of ESDCTL mask of the incoming DQS signal during read access.
- Write data DQS calibration—Set DQS output of each data byte to its delay relative to this byte’s DQ and DM outputs.
- Read data DQS calibration—Set DQS input of each data byte to its delay relative to this byte’s DQ inputs.
- Write data bit delay calibration—Additional delay for each output data bit.
- Read data bit delay calibration—Additional delay for each input data bit.
- Clock Signal delay calibration—Additional delay for the SDCLK and SDCLK_B signals.
- CA-Bus bit delay calibration—In LPDDR2 mode, additional delay is available for each bit of the command/address CA-bus.

3 Using Calibration for the Different DDR Standards

3.1 DDR2

When configuring the i.MX53 for DDR2, all of the above calibration processes should be considered, with the exception of write leveling calibration and CA-Bus bit delay calibration, which are associated with other DDR standards.

Although ODT activation is not typical for DDR2, it should be noted that the ZQ calibration is recommended for setting the right VCC/2 bias for the DDR2 signals.

3.2 DDR3

When configuring the i.MX53 for DDR3, all of the above calibration processes should be considered, with the exception of CA-Bus bit delay calibration**,** which is associated with other DDR standards.

3.3 LPDDR2

When configuring the i.MX53 for LPDDR2, all of the above calibration processes should be considered, with the exception of DQS gating calibration and write leveling calibration, which is associated with other DDR standards.

4 Calibration Over Frequency Range

Calibration, as with any other aspect of the DDR setup, is frequency dependant. Changing DDR clock frequency requires running the various calibration sequences and obtaining a new set of delay values.

The empirical rule is that a DDR setup for a target frequency, including a measured set of delay values at that frequency, is expected to be stable at the frequency range of ±10% around this point.

5 Calibration and Chip Selects

The user has the option of populating DDR memory devices on chip select 0 (CS0), chip select 1 (CS1), or both chip selects. Hardware calibration sequences, as described in this document, only use CS0 domain. For DDR devices connected to CS1, the user should take the following step to get valid calibration results:

- CS1 DDR device placement and route should strictly mirror to the layout of the CS0 devices. This board design rule is important for a common set of delay values, as well as several other aspects of the DDR port stability. The delay values received from CS0 hardware calibration can be used for CS1 operation as well when the placement and routing on each chip select match each other.
- If calibrating explicitly for CS1 is still required (for example, DDRs are populated only on CS1), the user can apply the software calibration sequence, making sure the addresses used in the software flow are for the CS1 domain.

6 Source of Calibration Delay Values

Each of the Calibration parameters of Section 2, "DDR Calibration Modes," is supported by register(s) in ESDCTL that hold its delay values. The source of delay values might be:

- Analysis of board features and layout.
- Trial and error run of different delay values, using a selected DDR test pattern.
- Iterative calibration sequence to select the best delay values, based on ESDCTL logics.

While first two methods are supported for all calibration parameters, the iterative calibration sequence is only provided for:

- ZQ calibration
- Write leveling calibration
- DQS gating calibration
- Read data DQS calibration
- Write data DQS calibration

7 Using the DDR Calibration Modes

7.1 Calibration Usage General Notes

- The calibration sequence should be executed after the DDR memory has been initialized.
- The calibration sequence should be executed by code mounted on the internal RAM of the i.MX and not on the external DDR device, as calibration sequence activity might interfere with program fetch.

7.2 ZQ Calibration Usage

Variations of the ZQ resistor values are mainly due to IC process and environmental (temperature and voltage) variations. Repeating auto-initialized ZQ calibration sequence by i.MX53 logic is thus a must for proper DDR operation, and should be completed by a one-time forced ZQ calibration, as described in the following sections.

7.2.1 Calibrating Local ZQ Versus DDR Device ZQ

ZQ calibration is done in both i.MX53 DDR PHY and the DDR device. Control bits ZQ_MODE[1:0] should be set in order to determine the activation events of the local and distant ZQ resistors.

7.2.2 One-Time Forced Hardware ZQ Calibration

Both local i.MX53 DDR PHY and the DDR device ZQ calibrations can be performed as a one-time event, by setting a bit in ESDCTL register.

One-time hardware ZQ calibration mode is recommended as part of the DDR setup before initiating any other DDR activity. In particular, forcing a one-time ZQ calibration and waiting its completion, is required before any other calibration process describe in this document.

7.2.3 Auto Initialized ZQ Hardware Calibration

In addition, the local i.MX53 DDR PHY and the DDR device ZQ hardware controlled calibrations can be auto initialized by the ESDCTL, upon the following events:

- i.MX53 power up
- Exiting DDR self refresh mode
- Exiting slow precharge power down
- Periodic ZQ calibration sequence, with period time configured to ESDCTL register

ZQ calibration auto initialization mode is recommended for keeping the ZQ registers calibrated during i.MX53 operating modes and ambient temperature changes.

7.2.4 DDR Device ZQ Hardware Calibration Type

While i.MX53 DDR PHY auto initiated hardware calibration sequence is the same in all the above cases. DDR device sequence differs between long and short procedures, as described in Section 10, “ZQ Calibration.”

7.2.5 One-Time Software ZQ Calibration

Local i.MX53 DDR PHY calibration can also be performed as a one-time event, by writing selected ZQ values to the ESDCTL register, reading the ZQ comparator results, changing the ZQ values, and repeating the process until the comparator toggling point is detected.

- One-time software ZQ calibration does not apply to the DDR device.
- One-time software ZQ calibration mode is mainly used for ESDCTL debugging. Keeping ZQ calibration in functional mode should be done by the other ZQ calibration modes.

7.3 Timing Calibrations Usage

The user can calibrate DDR timings (DQS gating, Write leveling and Write/Read DQS delay calibrations) using the DDR controller iterative calibration sequence features. Alternately, user can select a previously defined set of timing delay values and write them to delay registers, without calibration sequence activation. Not using any of these two methods, DDR controller is interfacing DDR device with its default (mid-range) delay values.

DDR timing variations are mainly associated with board design and selected DDR device type.

To a lesser extent, DDR timing are influenced by parametric deviations between manufactured i.MX53 ICs, DDR devices, and boards.

Changing ambient temperature and product age can also have a minor effect on DDR timing.

Considering all of these, user can apply any of the following calibration policies:

- Run timing calibration sequence once for certain board design. Reuse calibration values result as fixed delay values for all the population of this board type and the ICs assembled.
- Timing calibration sequence run on each and every manufactured board + ICs, as an additional assembly line procedure. The resulting delay values can be then flashed to the product and be used as delay preset values in DDR setup throughout its lifetime.
- Product software repeatedly runs calibration sequences upon system power up.

There is no known need to repeat the timing calibration process during system activity.

7.4 Order of Running the Calibration Sequence

When running timing calibration sequence, the following order should be kept:

1. One-time, forced ZQ Calibration
2. Write Leveling Calibration
3. DQS Gating
4. Read data DQS calibrations
5. Write data DQS calibrations

8 Delay Unit Hardware Overview

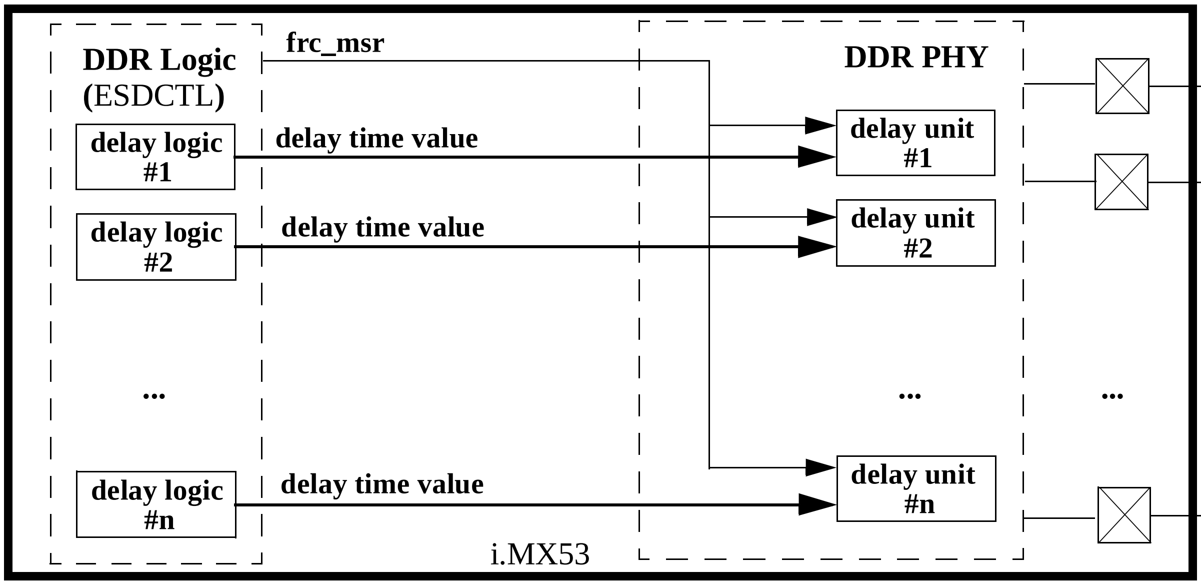

Figure 1. i.MX53 Delay Units Hardware: Functional Diagram

There are several different delay mechanisms in ESDCTL. Each of them is duplicated per byte or bit, as described throughout this document. This section in particular describes the general hardware structure and functionality which is in common to them all.

The delay mechanism is split into two parts:

- Delay logic, located at the ESDCTL, maintains the delay value in time units, and in some cases also handle a delay calibration sequence.
- Delay unit, located at the DDR PHY, contains a physical chain of basic delay elements.

DDR PHY supports an ongoing measurement process, to determine what is the time delay of the basic delay element. This basic time delay varies over temperature, and IC manufacturing. So this ongoing measurement is necessary.

A major signal in the handshake between delay logic and delay unit is the internal signal frc_msr.

“frc_msr,” when sent by delay logic, after delay time value is updated, drives the following activities:

- Absolute delay time value from delay logic is latched inside DDR PHY.
- DDR PHY uses latest measured process to determine how many delay elements should be activated to achieve the required absolute delay time value, and set the delay unit accordingly.

“frc_msr” must be asserted at the end of a delay calibration. In the auto-initiate and hardware calibration sequence cases, hardware asserts this signal.

In software-controlled and forced delay value calibration cases, it is the user’s responsibility to assert this signal, by setting the FRC_MSR bit of the ESDCTL_MUR register, at the end of the calibration sequence.

NOTE

In previous i.MX products, there was a dual delay calibration mechanism, allowing the user to update the delay by programming the absolute time delay value, or by number of delay elements.

In ESDCTL, the latter is no longer available. Only absolute time delay value programming is possible for the user. However, there are read only status registers which allow the user to read from the DDR PHY the calculated number of delay elements:

- WLDLST register for Write leveling
- DGDLST register for DQS gating

Reading the delay unit numbers is mainly used for IC validation and factory tests. For the ESDCTL user, this is not a valuable source of information.

9 Address and Data Content Used for Timing Calibration

Read DQS delay, Write DQS delay and DQS gating calibration sequences all use the same address, data content and compare registers when checking different delay values.

9.1 Address Used in Timing Delay Calibration Sequence

The base address used during read and write operations are hard coded to be (Bank0, Row0, Col0).

In i.MX53, this is mapped to CSD0_BASE_ADDR +0x10000000, or absolute address of 0x80000000.

Before any of the above calibration sequences take place, the user should fill this target address with content matching ESDCTL_PDCMPR1 (i= 0 to 3):

- PDV1[7:0] to [CSD0_DDR_BA+ 0x10000000 + i * 16]
- PDV1[15:8] to [CSD0_DDR_BA+ 0x10000004 + i * 16]
- PDV2[7:0] to [CSD0_DDR_BA+ 0x100000008 + i * 16]
- PDV2[15:8] to [CSD0_DDR_BA+ 0x1000000c + i * 16]

NOTE

Looking into ESDCTL programming model, the user can see a description of a “dummy write and read” mechanism. This mechanism, is required for systems where (Bank0, Row0, Col0) address is blocked from explicit core access. The ESDCTL accesses this address with no system read or write, and sets the data content as required for the calibration compare.

In i.MX53, however, the (Bank0, Row0, Col0) can be explicitly accessed. This entire document describes setting content of (Bank0, Row0, Col0) with explicit access, ignoring the “dummy access” option.

9.2 Optimal Data Content for DQS delay Calibration Sequence

- PDV2[15:8] is compared with all 8 data bytes of the 4th and 8th DQS (rising) edges of the calibrating burst
- PDV2[7:0] is compared with all 8 data bytes of the 3rd and 7th DQS (falling) edges of the calibrating burst
- PDV1[15:8] is compared with all 8 data bytes of the 2nd and 6th DQS (rising) edges of the calibrating burst
- PDV1[7:0] is compared with all 8 data bytes of the 1st and 5th DQS (falling) edges of the calibrating burst

Users are free to program any value to ESDCTL_PDCMPR1. There are, however, two rules for selecting an effective value:

1. Data should be toggled from rising edge to rising edge, and from falling edge to falling edge (data is separated at IO into two streams according to edge).
2. Better toggle all bits at the same direction.

Following these rules, ESDCTL_PDCMPR1 was set to 0x00ffff00, with some systems, this was empirically found to give the best calibration results.

10 ZQ Calibration

ZQ calibration is a process that tunes the DRAM and ESDCTL I/O pad output drivers (drive strength) and ODT values across changes in process, voltage, and temperature. There are two instances where ZQ calibration is performed: on the i.MX53 (DDR pads) and on the DDR device. ZQ calibration sequence can be initiated as a one-time event, or a periodic sequence.

One-time ZQ calibration can be one of the following two:

- One-time forced hardware ZQ calibration iterative sequence, as described in detail in Section 10.1, “One-Time Forced Hardware ZQ Calibration Sequence.”
- One-time software handled calibration iterative sequence ZQ calibration, as described in detail in Section 10.2, “One-time Software ZQ Calibration Sequence.”

Once periodic ZQ calibration (described in detail in Section 10.3, “Auto Initiated Hardware ZQ Calibration Sequence”) is programmed, the ESDCTL will automatically issue and perform the ZQ calibration without further user interaction. While i.MX53 side periodic calibration is the same at all times, there are two different types of ZQ calibration commands (long and short) that are sent to the DDR device and are described as follows.

Long DDR device ZQ calibration is initiated upon:

- One time forced ZQ calibration
- i.MX53 power up
- Exiting DDR self refresh mode
- Exiting slow precharge power down

Short DDR device ZQ calibration is initiated when a periodic ZQ is issued.

10.1 One-Time Forced Hardware ZQ Calibration Sequence

One-time ZQ calibration, using hardware loop logic is performed by the following sequence:

1. Make sure all accesses to DDR are finished.
2. Set ESDCTL register fields according to Table 1:

Table 1. One-Time Hardware ZQ Calibration Configurations

Register	Field	Description
ESDCTL Registers	--	Program the entire i.MX53 register set to fit the selected DDR mode and DDR device. It is recommended to “DDR INIT” setup code provided with the development kit or operating system in use.
ZQHWCTRL	ZQ_MODE[1:0]	One time hardware ZQ calibration involves both local ESDCTL PHY and DDR device. ZQ_MODE should thus be set to one of the following modes: • 0x1 ZQ calibration is issued to i.MX ZQ pad and external device (only when exiting self refresh). • 0x3 ZQ calibration is issued to i.MX ZQ pad and external device (both periodic and when exiting self refresh).
ZQHWCTRL	TZQ_INIT[2:0] TZQ_OPER[2:0]	For DDR2, DDR3 operation: • Set the DDR device ZQ timings to match DDR device Datasheet. • This field is ignored if DDR device ZQ calibration is not selected by ZQ_MODE.
ZQLP2CTL	ZQ_LP2_HWZQ INIT[8:0] ZQ_LP2_HW_Z QCL[7:0]	For LPDDR2 operation: • Set the DDR device ZQ timings to match DDR device Datasheet. • ESDCTL logic selects between ZQHWCTRL and ZQLP2CTL parameters according to ESDMISC/DDR_TYPE.
ZQHWCTRL	ZQ_PARA_EN	Device ZQ calibration parallel enable. 0 Device ZQ calibration is done in serial (CSD0 first and then CSD1). 1 ZQ calibration of both CS is done in parallel In functional mode, parallel calibration should be preferred for its speed. Choose serial as a for debugging if ZQ calibration issues are suspected.
ZQSWCTRL	USE_ZQ_SW_ VAL	Should be selected to ‘0’ for fields ZQ_HW_PD_VAL and ZQ_HW_PU_VAL to be used.

3. Assert the ZQ_HW_FOR bit in ZQHWCTRL register
4. Wait for ZQ_HW_FOR to be cleared by hardware
5. ZQ resistors are now calibrated. Pull-up, pull-down configured values can be read from ZQ_HW_PD_RES and ZQ_HW_PU_RES fields of the ZQHWCTRL register.

10.2 One-time Software ZQ Calibration Sequence

One time software ZQ calibration is mainly used to debug, and not recommended to be used with customer product calibration. One time software ZQ calibration only applies to local ESDCTL PHY and not for the

DDR device. It uses the hardware ZQ comparator, but the iterative sequence is handled by software. The following sequence should be followed:

1. Make sure all accesses to DDR are finished
2. Set ESDCTL register fields according to Table 2:

Table 2. One-Time Hardware ZQ Calibration Configurations

Register	Field	Description
ESDCTL Registers	—	Program the entire ESDCTL register set to fit the selected DDR mode and DDR device. It is recommended to “DDR INIT” setup code provided with the development kit or operating system in use.
ZQHWCTRL	ZQ_MODE[1:0]	One time software ZQ calibration involves Only local ESDCTL PHY. ZQ_MODE field options are not relevant in this case. Anyway, ZQ_MODE better be kept in one of the following modes: • 0x1 ZQ calibration is issued to i.MX ZQ pad and external device (only when exiting self refresh). • 0x3 ZQ calibration is issued to i.MX ZQ pad and external device (both periodic and when exiting self refresh).
ZQSWCTRL	USE_ZQ_SW_VAL	Should be selected to ‘1’ for fields ZQ_SW_PD_VAL and ZQ_SW_PU_VAL to be used.

10.2.1 Pull-up Calibration

1. Set ZQ_SW_PD in register ZQSWCTRL to ‘0’ (PU calibration)
2. Set ZQ_SW_PU_VAL in register ZQSWCTRL to a selected value
3. Assert the ZQ_SW_FOR bit in ZQSWCTRL register
4. Wait for ZQ_SW_FOR to be cleared by hardware
5. If ZQ_SW_RES is ‘1’, increase ZQ_SW_PU_VAL and repeat previous steps
6. When ZQ_SW_RES==0 is detected, stop sequence and read ZQ_SW_PU_VAL

10.2.2 Pull-Down Calibration

1. Set ZQ_SW_PD in register ZQSWCTRL to ‘1’ (PD calibration)
2. Set ZQ_SW_PD_VAL in register ZQSWCTRL to a selected value
3. Assert the ZQ_SW_FOR bit in ZQSWCTRL register
4. Wait for ZQ_SW_FOR to be cleared by hardware
5. If ZQ_SW_RES is ‘1’, decrease ZQ_SW_PD_VAL and repeat previous steps.
6. When ZQ_SW_RES==0 is detected, stop sequence and read ZQ_SW_PD_VAL

10.2.3 Apply ZQ Calibration

Set the FRC_MSR for the new calculated ZQ_SW_PU_RES and ZQ_SW_PD_RES start being used.

10.3 Auto Initiated Hardware ZQ Calibration Sequence

Auto initiated ZQ calibration, using hardware loop logic is performed by the following sequence:

1. Make sure all accesses to DDR are finished.
2. Set ESDCTL register fields according to Table 3:

Table 3. Auto Initiated Hardware ZQ Calibration Configurations

Register	Field	Description
ESDCTL Registers	--	Program the entire ESDCTL register set to fit the selected DDR mode and DDR device. It is recommended to “DDR INIT” setup code provided with the development kit or operating system in use.
ZQHWCTRL	ZQ_MODE[1:0]	Enables the automatic ZQ calibration process. It is recommended for the user to set this field to 0x3, which programs the ESDCTL to issue ZQ calibration to the i.MX ZQ calibration pad together with the ZQ calibration command to the external DDR device periodically (short command) and when exiting self refresh (long command).
ZQHWCTRL	ZQ_HW_PER[3:0]	ZQ hardware calibration period time. This field determines how often ZQ hardware calibration time is performed in regular operation mode. It is recommended for the user to simply leave this filed set to 1 ms (bit setting 0000). This field is ignored if periodic ZQ calibration is not selected by ZQ_MODE.
ZQHWCTRL	ZQ_HW_FOR	Force ZQ automatic calibration process with the i.MX ZQ calibration pad. No need to set this bit during normal operations as the ZQ calibration of the i.MX DDR pads are taken care of periodically.
ZQHWCTRL	TZQ_INIT[2:0]	TZQ_INIT holds the number of cycles that are required by the external DDR device to perform ZQ long calibration right after reset. The default bit setting of 100 (512 cycles) is recommended.
ZQHWCTRL	TZQ_OPER[2:0]	Holds the number of cycles that are required by the external DDR device to perform ZQ long calibration except the first ZQ long command that is issued after reset. The default bit setting of 011 (256 cycles) is recommended.
ZQHWCTRL	TZQ_CS[2:0]	Holds the number of cycles that are required by the external DDR device to perform ZQ short calibration. Note that the default value of 128 cycles (bit setting 010) is also the minimum amount of cycles that can be set. Therefore, if the external DDR3 device specifies this.
ZQLP2CTL	ZQ_LP2_HWZQINIT[8:0] ZQ_LP2_HW_ZQCL[7:0]	For LPDDR2 operation: • Set the DDR device ZQ timings to match DDR device Datasheet. • ESDCTL logic selects between ZQHWCTRL and ZQLP2CTL parameters according to ESDMISC/DDR_TYPE.
ZQHWCTRL	ZQ_PARA_EN	Device ZQ calibration parallel enable. 0- Device ZQ calibration is done in serial (CSD0 first and then CSD1) 1- ZQ calibration of both CS is done in parallel In functional mode, parallel calibration should be preferred for its speed. Choose serial as a for debugging if ZQ calibration issues are suspected.
ZQSWCTRL	USE_ZQ_SW_VAL	Should be selected to ‘0’ for fields ZQ_HW_PD_VAL and ZQ_HW_PU_VAL to be used.

3. ZQ resistors will be auto calibrated according to the operation mode selected by ZQ_MODE[1:0].

10.4 Local ESDCTL PHY Hardware ZQ Calibration Sequence

On forced or auto-initiated hardware calibration of local PHY, the sequence described below is done on the DDR_CALIBRATION pin by the ESDCTL logic (all steps are hardware driven—no user activity is required). ZQ calibration is terminated by latching the new PU and PD values to the entire DDR pads.

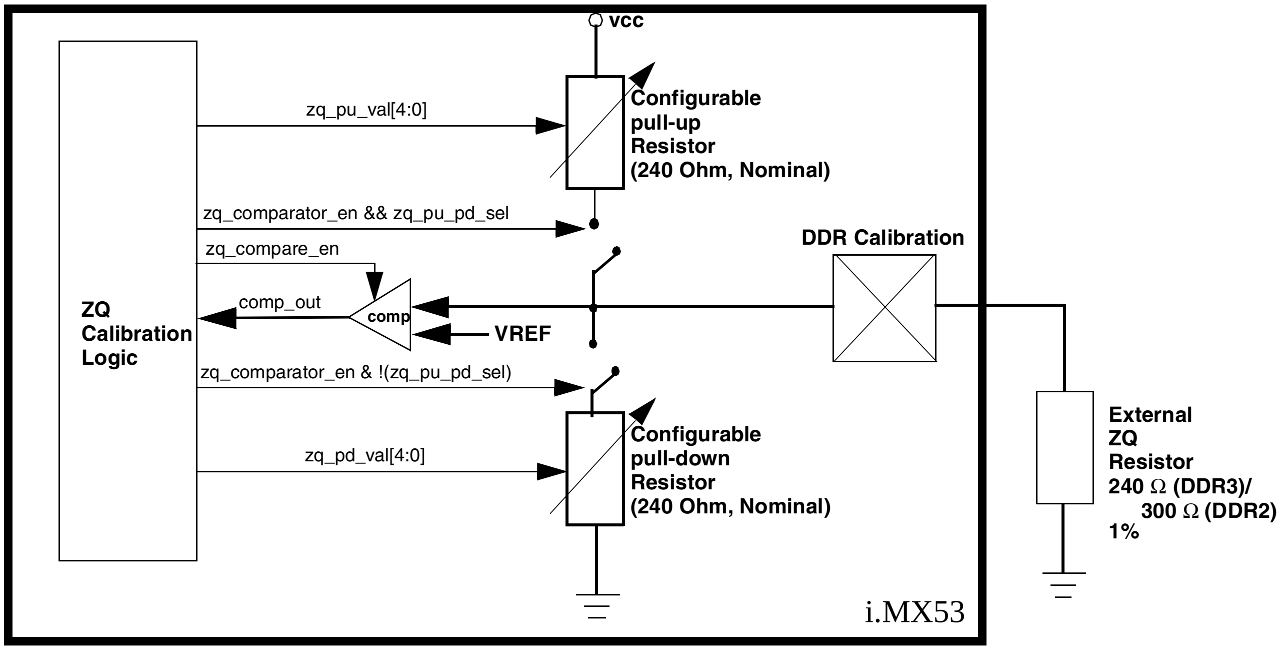
vcc
zq_pu_val[4:0]
Configurable pull-up Resistor (240 Ohm, Nominal)
zq_comparator_en && zq_pu_pd_sel
zq_compare_en
DDR Calibration
comp_out
comp
VREF
ZQ Calibration Logic
zq_comparator_en & !(zq_pu_pd_sel)
zq_pd_val[4:0]
Configurable pull-down Resistor (240 Ohm, Nominal)
External ZQ Resistor 240 Ω (DDR3)/ 300 Ω (DDR2) 1%
i.MX53

Figure 2. i.MX53 DDR PHY ZQ Calibration Hardware: Functional Diagram

10.4.1 Pull-Up Calibration (Step 1-8)

The internal pull-up resistor calibration is done using the 1% external resistor.

Step 1: zq_comperator_en and zq_pu_pd_sel are set to ‘1’.

Step 2: Set MSB of zq_pu_val (zq_pu_val=’10000’). If comparator output is 1 (output voltage is higher than Vdd/2 => internal resistor is less than external resistor) keep zq_pu_val[4] set. Else, clear zq_pu_val[4].

Step 3: Set Bit 3 of zq_pu_val (zq_pu_val=’x1000’). If comparator output is 1(output voltage is higher than Vdd/2 => internal resistor is less than external resistor) keep zq_pu_val[3] set. Else, clear zq_pu_val[3].

Steps 4–6: Repeat the same for bit 2 (step 4), bit 1 (step 5) and bit 0 (step 6).

Step 7: hw_zq_pu_en is driven to 0.

Step 8: ESDCTL drives ZQ calibration result to ZQHWCTRL[ZQ_HW_PU_RES].

10.4.2 Pull-Down Calibration (Step 9-16)

The internal pull-down resistor calibration is done versus the internal pull-up resistor.

Step 9: zq_comperator_en is set to ‘1.’ zq_pu_pd_sel is set to ‘0.’

Step 10: Set MSB of zq_pd_val (zq_pd_val=’10000’). If comparator output is 1 (output voltage is higher than Vdd/2 => internal pd resistor is less than external resistor) clear zq_pd_val[4] set. Else, keep zq_pd_val[4] set.

Step 11: Set MSB of zq_pd_val (zq_pd_val=’x1000’). If comparator output is 1(output voltage is higher than Vdd/2 => internal pd resistor is less than external resistor) clear zq_pd_val[3] set. Else, keep zq_pd_val[3] set.

Step 12: Repeat the same for bit 2

Step 13: Repeat the same for bit 1

Step 14: Repeat the same for bit 0

Step 15: hw_zq_pu_en is driven to 0

Step 16: ESDCTL drives ZQ calibration result to ZQHWCTRL[ZQ_HW_PD_RES].

10.4.3 Apply ZQ Calibration (Step 17)

Step 17: Set an internal “Force measure” signal for the new calculated ZQ_HW_PU_RES and ZQ_HW_PD_RES start being used.

10.5 DDR Device ZQ Calibration Sequence

Upon forced or auto initiated hardware calibration of DDR device occurs, the following sequence is performed by the ESDCTL logic (all steps are hardware driven, no user activity is required):

1. Send a precharge-all command to DDR device.
2. Wait for tRP period.
3. Send a ZQ calibration command to DDR device. CSD0/CSD1 are set according to selected DDR target. A10 is ‘0’ for long ZQ calibration, or ‘0’ for short ZQ calibration.
4. Keep the DDR port idle for the time period indicated by ZQ_HW_ZQC/ZQ_LP2_HW_ZQCL/ZQ_LP2_HW_ZQCS (according to DDR_TYPE).

11 Write Leveling

Write leveling is associated with the DDR “fly-by” board topology. “Fly-by” has its advantages of layout simplicity and minimal stubs. In the fly-by topology layout, the address, command, and clock signals are “daisy chain” routed from one DDR3 device to the next, where DQS and data (DQ) bus signals are routed “point-to-point” from the ESDCTL to the DDR3 device.

A skew is implied, however, between CK and DQS, a skew that is not the same for the different DQSx signals, depending on the board placement of the DDR device which is connected to the specific DQSx. See Figure 3.

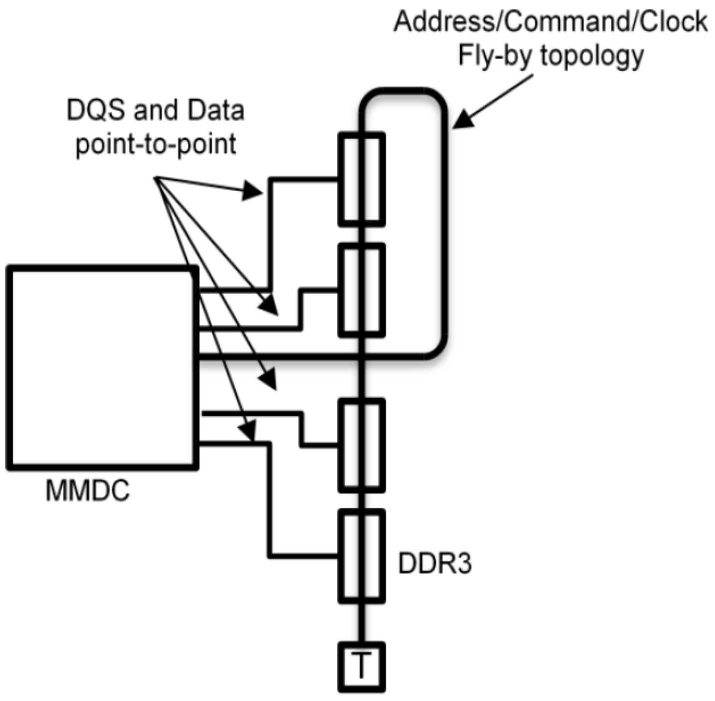

Figure 3. Fly-by Topology

The point of write leveling is to give each set of {DQSx, DMx, DQx} its own delay, to de-skew the clock to DQS timing relationship at the DDR3 device. A proper 'fly-by' design is based on clock, address and command trace length, which is equal or greater than any of the data bytes trace lengths.

ESDCTL write leveling is based on this, as its delay value compensates for early data signals, while late data signals can't be compensated.

Write leveling calibration is a process that involves both the ESDCTL and the DDR device. The WL process has its own DDR signal setup which is different than DDR functional mode. During WL, DQSx is constantly driven by i.MX53. DQx is constantly driven by the DDR device.

WL sequence is described in Figure 4. i.MX53 repeatedly sends single strobes on DQSx. CK is continuously sampled by DDR on DQSx rising edge and feed back the result to DQx. When CK and DQSx are unaligned, returned value is ‘0’. i.MX53 then adds delay to DQSx, until ‘1’ is returned on DQx. The measured delay is then applied to the entire {DQSx, DMx, DQx} group.

Each DQS has its own delay. In DDR3, the configurable value for each DQS can be different. WL calibration sequences of the different DQSx are done in parallel.

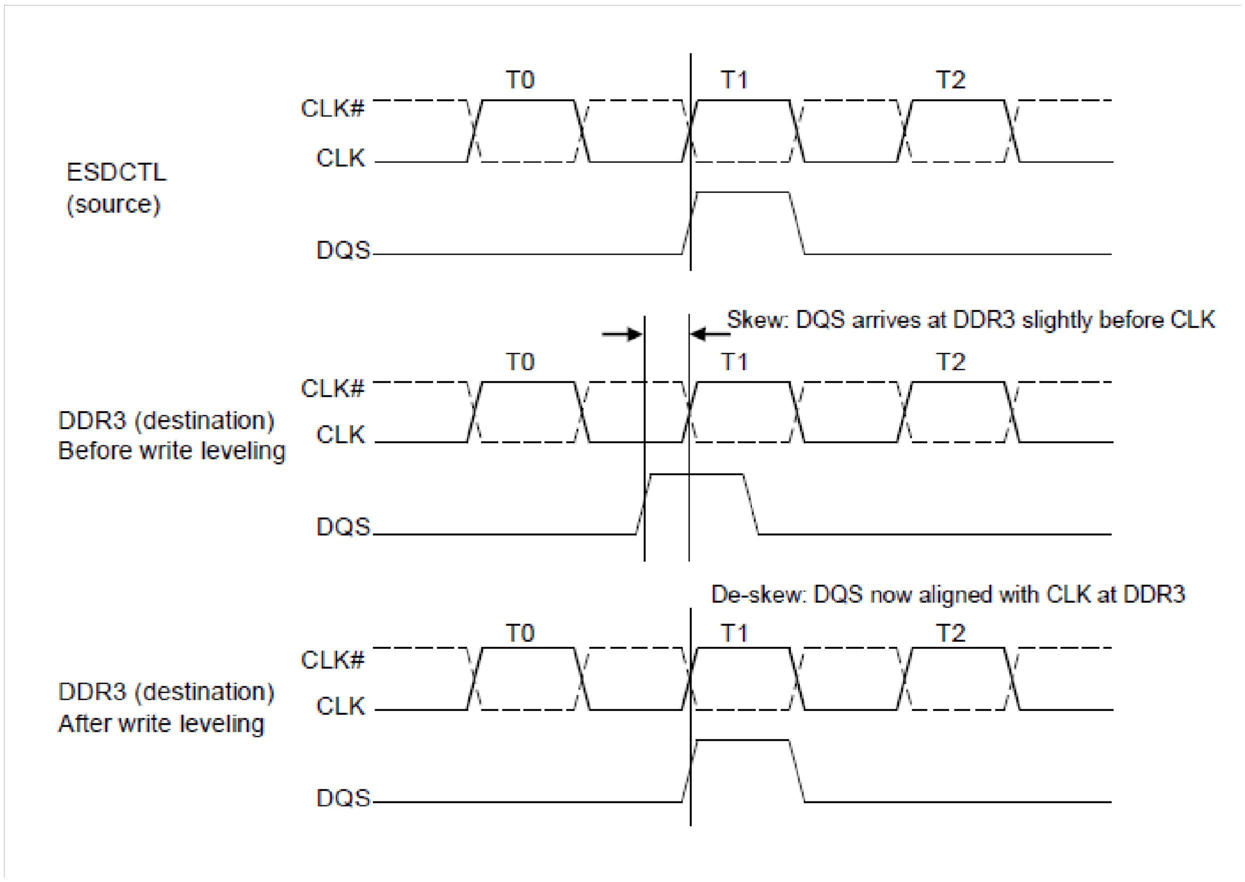

Figure 4. Timing Diagram: Before and After the Write Leveling

The i.MX53 supports up to 2.875 cycles of physical delay. Delay value is combined from the WL delay fields: (WL_DL_ABS_OFFSET/256 * cycle) + (WL_HC_DEL * half cycle) + (WL_CYC_DEL * cycle).

It should be noted, however, that the WL calibration, as defined for the i.MX53 and the DDR device is limited to detect and align CK to DQSx skews of up to one cycles. Larger skews will cause DQSx to overlap and detect the next CK cycle, returning fault alignment feedback. Good DDR board design can keep the skew smaller than 1 cycle. If, however, a larger skew is required, user can program the number of integer cycle delays to the WL_CYC_DELx in WLDECTRLx register, and the WL calibration will be modified accordingly.

It should be noted that write leveling delay might also imply an increase of the overall cycle time of the DDR write access. This increase is achieved by changing WALAT parameter from its default 0 to 1 (or more, in extreme cases). To be on the safer side, WALAT is always increased during the WL calibration process (see Section 11.2, “Hardware Write Leveling Calibration Sequence”). If the resulting WL delays are small, WALAT can be written back to 0, for the DDR normal operation. If, on the other hand, WL delays are significant (empirically seen to be at about 1/2 cycle and above, for one of DDR bytes, or more), increased WALAT should be kept as part of the new DDR setup that include the newly calculated WL delays. Increased WALAT implies a small degradation in DDR port performance.

It should be noted that the entire WL calibration is done using CK0, with DDR devices connected to CSD0. Applying CSD0 delay results to the DDR devices connected to CSD1 requires a DDR board design that provides an identical route and placement to CSD0 and CSD1 DDR devices, sharing the same DQx bytes (typically done by mirror design of CSD0 and CSD1 DDR placement using both sides of the PCB).

i.MX53 WL calibration is sensing LSB of the DQx byte for the WL feedback. That is, bits 0, 8, 16, and 24 of the DQ bus are being used. This fact should be considered during board design. Regularly, board design is allowed to swap bit connections within the DDR byte, wherever it helps DDR route optimization. Swapping DQx LSBs, as above, will disable operation of the WL calibration. Swapping the LSBs should be avoided, unless a DDR device that reflect WL feedback to all bits of the DQx byte is used.

11.1 Calibrating Write Leveling with a Preset Delay Value

Write leveling can be calibrated without running the calibration sequence by programming a predefined value of absolute delay.

The source of the predefined value can be a write leveling calibration sequence done in the past. Alternately, the delay values can be an estimation of data delays based on DDR board route length.

Based on simulations and experience with boards, the following empirical rule can be used:

Each ([SDCLK_LENGTH] -[DQS_LENGTH])/6, measured in inches, implies 1 ns of delay.

The time delay received by this rule should be converted to ddr_cycle/256 units, and applied to delay register.

Calibrating with a preset value is done by the following:

- Write the WL preset value to WLDECTRL0, WLDECTRL1 registers
- Set MUR[FRC_MSR] bit.

11.2 Hardware Write Leveling Calibration Sequence

The following steps should be executed by user code:

1. Store the contents originally programmed in the DDR3 MR1 register. This register will be overwritten in step 3 to enter write leveling mode (DDR3 device does not allow read of the mode registers, so MR1 content cannot be retrieved by read). Refer to the DDR3 device data sheet for details on the MR1 register.
2. Disable Auto initiated ZQ calibration and DDR auto refresh, so these processes would not interfere with the WL calibration.
3. Increase WALAT and RALAT parameters to maximum.
4. Configure the external DDR device to enter write leveling mode through MRS command.

5. Activate the DQS output enable by setting ESDSCR[WL_EN]. Note that current and previous steps can be performed simultaneously as both steps involve writing to the same ESDSCR register.

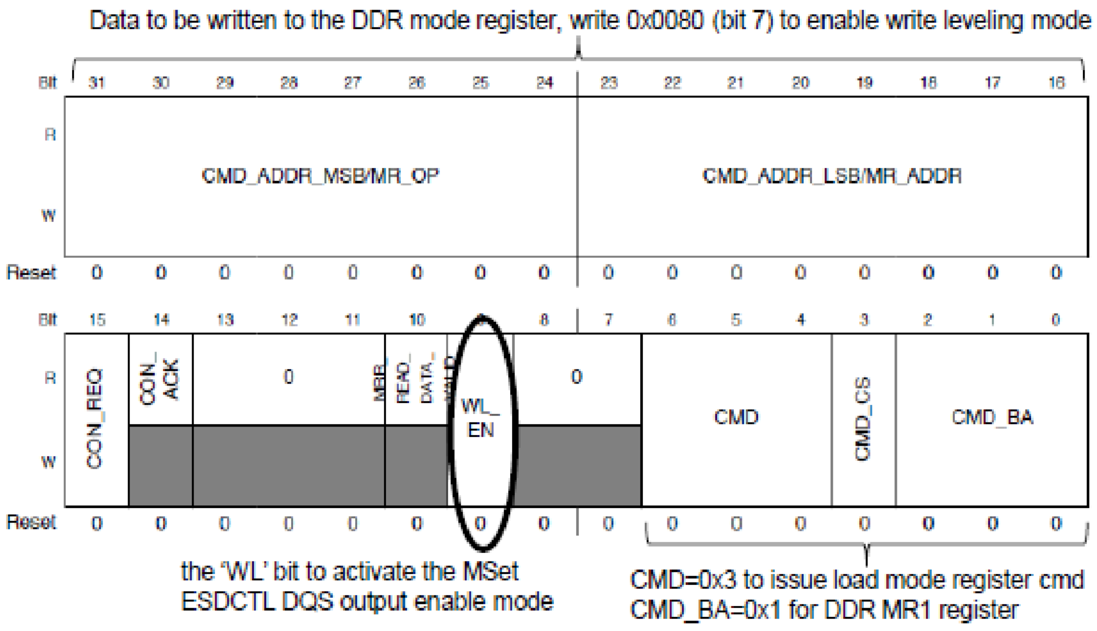
Data to be written to the DDR mode register, write 0x0080 (bit 7) to enable write leveling mode
Bit
R
W
Reset
CMD_ADDR_MSB/MR_OP
CMD_ADDR_LSB/MR_ADDR
CON_REQ
CON_ACK
MRR_READ_DATA_VALID
WL_EN
CMD
CMD_CS
CMD_BA
the 'WL' bit to activate the MSet ESDCTL DQS output enable mode
CMD=0x3 to issue load mode register cmd
CMD_BA=0x1 for DDR MR1 register

Figure 5. ESDSCR Register Example to Enable Write Leveling

6. Activate automatic calibration by setting WLGCR[HW_WL_EN].
7. Poll the WLGCR[HW_WL_EN] until cleared.
8. Read WL error bits, WLGCR[11:8], to verify WL calibration sequence terminated properly.
9. Configure the external DDR device to leave write leveling mode through MRS command (Write to DDR3 mode register MR1, clear bit 7, through the ESDSCR).
10. Clear ESDSCR[WL_EN]. As described above, current and previous steps can be performed simultaneously as both steps involve writing to the ESDSCR register.
11. Configure ESDCTL to functional mode DDR parameters: RALAT, WALAT, auto ZQ calibration and auto refresh.
12. Once this process is done, the WLDECTRL0 and WLDECTRL1 registers are updated with the proper write leveling delays. These values can be kept and reused in the future, as described in Section 11.1, "Calibrating Write Leveling with a Preset Delay Value."

After WLGCR[HW_WL_EN] bit is activated (step 6 above), the following steps are executed automatically by the ESDCTL:

1. ESDCTL enters write leveling mode, counts 25 + 15 cycles and drives the DQS pads as output while the DQ pads will remain inputs. In parallel the ESDCTL configures the write leveling delay line to "0" (WLDECTRL0[WL_DL_ABS_OFFSETx] = 0) and issues a measurement process of the write-leveling delay-line to update itself with the new value.
2. ESDCTL drives one DQS pulse to the DDR external device.

3. ESDCTL waits 16 cycles (to guarantee that the DQ prime data is stable) and samples the associated prime DQ bit (for example for DQS1 the ESDCTL samples DQ[8]).
4. ESDCTL increments the write leveling delay line by 1/8 cycle and performs a measurement process in order to load the updated value to the associated delay-line.
5. ESDCTL repeats steps 5-7 till the write leveling delay is 1 cycle.
6. ESDCTL checks the 8 bit prime DQ results for each DQS and finds the first transition from 0 to 1. If no transition is found then the ESDCTL indicates an error at WLGCR[HW_WL_ERRx].
7. ESDCTL stores the value that issues the last “0” on the prime DQ before the transition and loads it to the write leveling delay-line. The ESDCTL initiates a fine-tune process by incrementing the delay-line values by 1 step (which is 1/256 part of a cycle) till detecting the most accurate transition from 0 to 1.
8. Upon completion of this process the ESDCTL de-asserts the WLGCR[HW_WL_EN] and updates the most accurate value of the delay-line at the associated. WLDECTRLx[WL_DL_ABS_OFFSETx]
9. ESDCTL performs a measurement process (internal frc_msr) in order to load the most accurate value to the associated delay-line.

12 DQS Gating

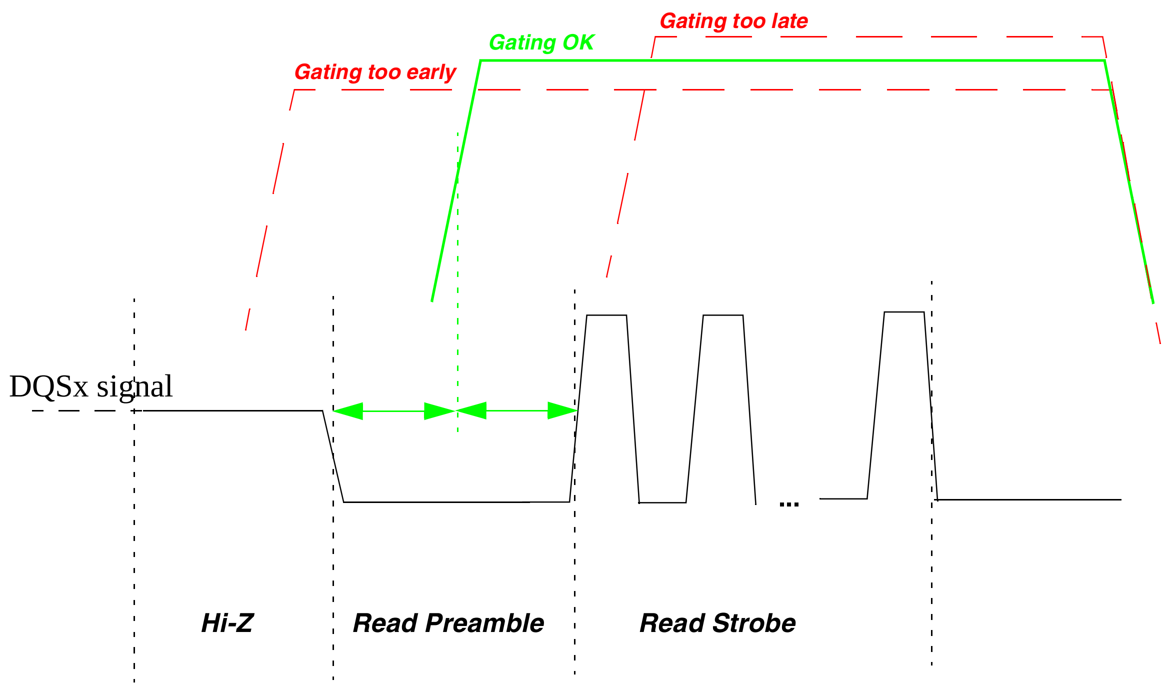

Figure 6. DQS Read Gating Calibration: Timing Diagram

The read DQS gating calibration is not mandatory by DDR3 JEDEC standard, but it is a required mechanism to determine the valid time period with which to begin sampling the incoming “read” DQS signal. The intent is to avoid inadvertently sampling an invalid DQS state, such as misinterpreting the initial High-Z to logic low transition as a valid read DQS edge.

- DQS gating is thus an over-time masking operation of the input DQS driven by the DDR device.
- DQS gating assertion is done according to the DQS gating delay, as configured to the ESDCTL registers.
- DQS gating negation has a fixed timing after the negative edge of the last DQS strobe (if there is no back to back access).

As shown in Figure 6, a properly set DQS gate is starting at the middle of the DQS read preamble, that is, at the middle of the low DQS driven by the DDR device ahead of the first DQS strobe. Too early DQS gating assertion causes ESDCTL to sense the DQS signal when floating, and possibly sense false DQS signal assertions. Too late DQS gating assertion masks a valid DQS strobes from being sensed by the ESDCTL.

DQS gating calibration is the process of giving the DQS gate the proper delay. DQS latency is changing according to DDR device and board design, so DQS gate delay should be measured for each unit separately.

The hardware DQS gating calibration sequence is based on finding the too-early and too-late gate boundaries, placing the boundary values at the status registers (DGHWSTx), then calculating the arithmetic average of the boundaries and placing it as delay value in the control registers (DGCTRLx).

During read DQS calibration, the ESDCTL samples only the DQS signal. Its complementary DQS_B signal is not part of the process. The assumption is that the signal pair is well balanced, thus sharing the same timing and require the same gate masking.

Each of the four DQS signals has a its own gating delay and gating calibration. The DQS gating calibration sequence, however, can be done for all DQS in parallel.

The DQS gating includes a delay of up to 7.875 cycles. Delay value is combined from the DQS gating delay fields: (DG_DL_ABS_OFFSET/256 * cycle) + (DG_HC_DEL * half cycle).

It should be noted that in order to mask out faulty incoming DQS strobes, there is an alternative to DQS gating, by simply applying pull downs to DQS signals. Pull-downs, however, have their disadvantages, which include excessive power consumption and interference with ODT resistors. DQS gating is free from these, and hence considered as a better choice.

12.1 Calibrating DQS Gating with a Preset Delay Value

DQS gating can be calibrated without running the calibration sequence, by programming a predefined value of absolute delay.

The source of the predefined value can be a DQS gating calibration sequence done in the past, or any other value estimated by user to optimize DQS gating location.

Calibrating with a preset value is done by the following:

1. Write the DQS gating preset value to DGCTRL0, DGCTRL1 registers
2. Set MUR[FRC_MSR] bit.

12.2 Hardware DQS Gating Calibration Sequence

- Options to set the data content used during DQS gating calibration:
  - Calibration with ESDCTL predefined data, written to a selected address in DDR device RAM.
- There are two options to handle the iterative DQS gating sequence:
  - Hardware DQS gating sequence
  - Software DQS gating sequence

  Hardware DQS gating calibration is the recommended sequence for the typical customer board, while software sequence is only used for debug and special cases, when necessary.

It should be noted that the entire DQS gating hardware calibration sequence is done using CK0, with DDR devices connected to CSD0. The CSD0 delay results can be applied to the DDR devices connected to CSD1, if the DDR board routing provides an identical routing and placement to both CSD0 and CSD1 DDR devices, where all share the same DQx bytes (typically done by mirror design of CSD0 and CSD1 DDR placement using both sides of the PCB).

During the DQS gating software calibration sequence, the user can select any address for the write and read operations so the devices connected to CSD1 can participate similarly to CSD0 devices.

12.3 ESDCTL Register Setup for DQS Gating Calibration Sequence

12.3.1 Register Setup Prior to DQS Gating Calibration Sequence

Set ESDCTL register fields according to Table 4:

Table 4. Hardware DQS Gating Calibration Configurations

Register	Field	Description
i.MX53 registers	--	Program the entire i.MX53 register set to fit the selected DDR mode and DDR device. It is recommended to “DDR INIT” setup code provided with the development kit or operating system in use.
IOMUXC/ SW_PAD_CTL _DQSx	pke, pue, pus	Configure DQS pads to have an active on-chip 47k pull-up. Pull up is required to make sure the too early DQS gating settings will drive false data, by failing to gate off the Hi-Z period.
DGCTRL0	DG_EXT_UP	DG extend upper boundary. ‘0’ - upper boundary of DQS gating hardware calibration is set according to first fail after at least one pass. ‘1’ - upper boundary is set according to the last pass. There is no significant difference observed in calibration results due to this bit setup. It typically remains clear.
DGCTRL0	DG_DIS	Keep the writable DG_DIS field cleared for the DQS gating to take place.
DGCTRL0	DG_CMP_CYC	‘0’ PHY compares the received data 16 cycles after sending the read command ‘1’ PHY compares the received data 32 cycles after sending the read command For final DQS gating sequence procedure, prefer the faster 16 cycle delay. Use the 32 cycle delay for initial operation and debugging of the DQS gating sequence.

12.3.2 Calibration Sequence Setup with Predefined Data Content

In the case where predefined mode is used—in other words, when (PDCMPR2[MPR_CMP]) is cleared—then the following steps should be executed:

1. Issue two resets to the read data FIFO through the DGCTRL0 register by writing to the RST_RD_FIFO bit (DGCTRL0[RST_RD_FIFO]=1) polling this bit to clear for completion of reset, and then repeat this sequence one more time. This sequence must be performed twice in order to properly reset the read data FIFO. Precharge all active banks (can be done through ESDSCR) as required by the standard.
2. Configure the predefined value, which reflects the value that will be written and compared through the read calibration, to PDCMPR1[PDV1, PDV2].
3. Issue write access to the external DDR device, with content matching [PDV1,PDV2]. For ESDCTL the following should be written (i= 0 to 3):
   — PDV1[7:0] to [CSD0_DDR_BA+ 0x10000000 + i * 16]
   — PDV1[15:8] to [CSD0_DDR_BA+ 0x10000004 + i * 16]
   — PDV2[7:0] to [CSD0_DDR_BA+ 0x100000008 + i * 16]
   — PDV2[15:8] to [CSD0_DDR_BA+ 0x1000000c + i * 16]

   [PDV1,PDV2] setup can be any value but empirical data shows that a predefined value of 0x00FFFF00 yields the most robust results.
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (RDDLCTL[RD_DL_ABS_OFFSET#]) will place the read DQS somewhere inside the read DQ window. Note that the default value of 0x40 should be sufficient so no action is necessary by the user, unless the user has changed these values prior to the calibration process.
5. Start the calibration process by asserting DGCTRL0[HW_DG_EN]. The ESDCTL will then perform the hardware sequence as described in Section 12.4, “DQS Gating Hardware Calibration Sequence.”
6. Poll the DGCTRL0[HW_DG_EN] until it is cleared to indicate the completion of the hardware calibration. Also, check DGCTRL0[HW_DG_ERR] to see if any errors occurred during calibration.
7. Result DQS gating delay values are now valid at DGCTRL0 and DGCTRL1 registers. Content can be read a saved for future delay setup. DQS gating window high and low edges are valid at DGHWST0, DGHWST1, DGHWST2 and DGHWST3.
8. Alternately, if software DQS gating is required, go on with the user code, as described in Section 12.5, “DQS gating Software Calibration Sequence.”

12.4 DQS Gating Hardware Calibration Sequence

The following steps will be executed automatically by the ESDCTL, after DGCTRL0[HW_DG_EN] is asserted:

1. ESDCTL waits till the read DQS delay-line is updated with the absolute delay value for all bytes at DGCTRLx[DG_HC_DELx] and DGCTRLx[DG_DL_ABS_OFFSETx] and also satisfying the Tmod + 4 requirement.

2. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
3. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point. If the comparison passes then ESDCTL advances to step 9.
4. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
5. ESDCTL increments the read DQS gating delay of each byte by half cycle (DGCTRLx[DG_HC_DELx] + 1).
6. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
7. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
8. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 5-8. If the comparison passes then ESDCTL stores the value of the temporary low boundary and advances to next step.
9. ESDCTL increments the read DQS gating delay-line of each byte by half cycle (DGCTRLx[DG_HC_DELx] + 1) and issues a measurement process of the read DQS gating delay-line to update itself with the new value.
10. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
11. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
12. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 9-11. If the comparison fails then ESDCTL stores the value of the temporary upper boundary and starts searching the adequate low and high boundaries.
13. ESDCTL returns to the temporary low boundary minus half cycle and issues a measurement process of the read DQS gating delay-line to update itself with the new value.
14. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
15. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
16. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 14-15. If the comparison passes then ESDCTL stores the value of the adequate low boundary and advances to step 24.
17. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
18. ESDCTL increments the read DQS gating delay of each byte by 1 (DGCTRLx[DG_DL_ABS_OFFSETx] + 1) and issues a measurement process of the read DQS gating delay-line to update itself with the new value and advances to step 19.

19. ESDCTL returns to the temporary upper boundary minus half cycle and issues a measurement process of the read DQS gating delay-line to update itself with the new value.
20. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
21. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
22. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 20-21. If the comparison fails then ESDCTL stores the value minus 1 of the adequate upper boundary and advances to step 23.
23. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
24. ESDCTL increments the read DQS gating delay of each byte by 1 (DGCTRLx[DG_DL_ABS_OFFSETx] + 1) and issues a measurement process of the read DQS gating delay-line to update itself with the new value and advances to step 25.
25. After the ESDCTL finds the window boundary (lower and upper) of each read data byte then it stores the average between lower and upper boundaries at the associated DGCTRLx[DG_HC_DELx, DG_DL_ABS_OFFSETx] and issues a measurement process of the read DQS delay-line to update itself with the new value.
26. In addition, for debug information, the upper and lower boundaries for each byte found during the calibration process can be found in the 4 status registers: DGHWSTx[HW_DG_UPx, HW_DG_LOWx].
27. ESDCTL indicates that the read DQS gating calibration had finished by setting DGCTRL0[HW_DG_EN] = 0.

12.5 DQS gating Software Calibration Sequence

The following steps should be executed by user code:

1. Precharge all active banks (can be done through ESDSCR) as required by the standard.
2. Configure the predefined value, which reflects the value that will be written and compared through the read calibration, to PDCMPR1[PDV1, PDV2].
3. Issue write access (with any legal DDR address) to external DDR device.
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (RDDLCTL[RD_DL_ABS_OFFSET#]) will place the read DQS somewhere inside the read DQ window.
5. Configure the read DQS delay-line to issue zero delay by setting DGCTRLx[DG_DL_ABS_OFFSETx] = 0 and DGCTRLx[DG_HC_DELx] = 0.
6. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.
7. Wait 16 DDR cycles till the read DQS delay-line is updated with the absolute delay value for all bytes.

8. Issue read command (with the legal DDR address chosen in step 3) from the external DDR device and wait 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
9. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point. If the comparison passes then advance to step 10.
10. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
11. Increment the read DQS gating delay of each byte by half cycle (DGCTRLx[DG_HC_DELx] + 1)
12. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
13. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 10-12. If the comparison passes then advance to step 14.
14. Store the temporary lower boundary and start searching the temporary upper boundary
15. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
16. Increment the read DQS gating delay of each byte by half cycle (DGCTRLx[DG_HC_DELx] + 1)
17. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
18. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
19. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 15-18. If the comparison fails then it is needed to store the value of the temporary upper boundary and starts searching the adequate low and high boundaries.
20. Load the temporary low boundary minus half cycle into the associated DGCTRLx[DG_HC_DELx].
21. Reset the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
22. Increment the read DQS gating delay of each byte by 1 (DGCTRLx[DG_DL_ABS_OFFSETx] + 1) and force the delay line to measure itself and to issue the requested read DQS delay by configuring MUR[FRC_MSR] = 1.
23. Issue read command to the external DDR devices and wait 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
24. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).

25. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 21-24. If the comparisons passes then advance to the next step.
26. Store the adequate lower boundary.
27. Load the temporary upper boundary minus half cycle into the associated DGCTRLx[DG_HC_DELx].
28. Reset the read FIFO (to the inverted predefined value) and it’s pointers by setting DGCTRL[RST_RD_FIFO] = 1.
29. Increment the read DQS gating delay of each byte by 1 (DGCTRLx[DG_DL_ABS_OFFSETx] + 1) and force the delay line to measure itself and to issue the requested read DQS delay by configuring MUR[FRC_MSR] = 1.
30. Issue read command to the external DDR devices and wait 16 or 32 cycles (according to DGCTRL0[DG_CMP_CYC]) assuming that the data has arrived from the DDR device.
31. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8).
32. If the comparison passes then it is needed to repeat steps 28-31. If the comparisons fails then advance to the next step.
33. Reset the read FIFO (to the inverted predefined value) and it’s pointers by setting DGCTRL[RST_RD_FIFO] = 1.
34. Store the adequate upper boundary.
35. Calculate the average between lower and upper boundaries of each read DQS gating value and stores the average at the associated DGCTRLx[DG_DL_ABS_OFFSETx].
36. Issue the requested read DQS delay by configuring MUR[FRC_MSR] = 1.

13 Read DQS Delay Calibration

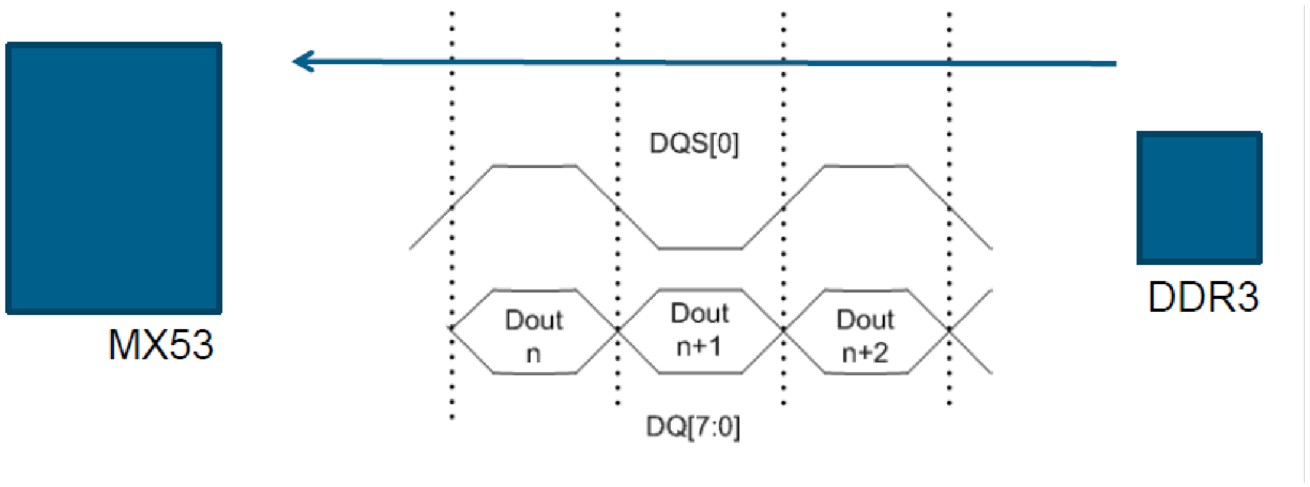

Figure 7. Data read illustration showing edge aligned DQS with DATA

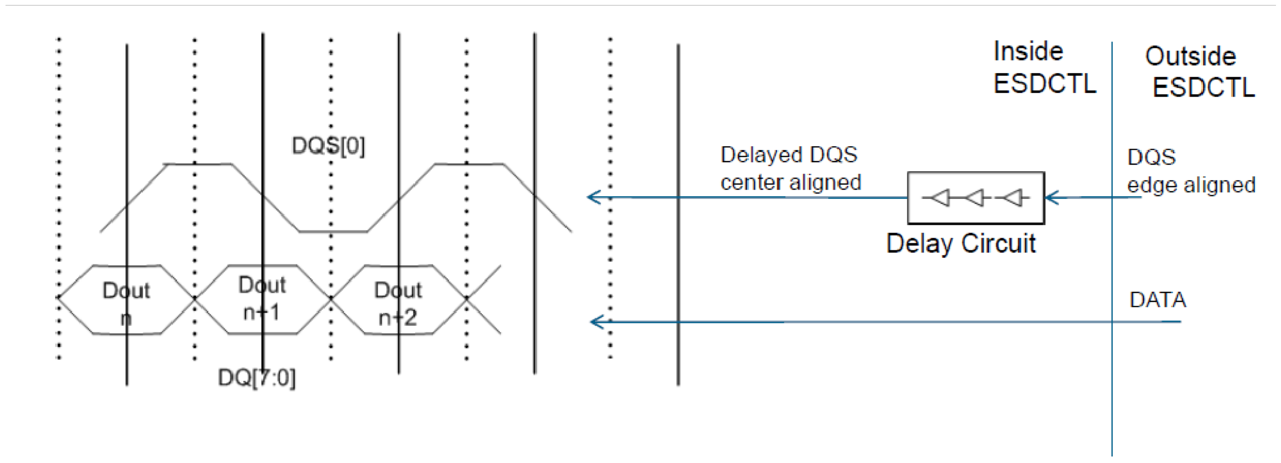

Figure 8. Illustration of DQS Being Delayed to Center Align it with Incoming Read Data

The read delay-line calibration is used to adjust the read-DQS within the read-data byte. When read-data is output from the DDR device, the associated read-DQS rising and falling edges are edge aligned with the data. When the read-DQS and read-data reach the ESDCTL, the ESDCTL in turn delays the sampling of DQS by ¼ cycle to center align the DQS edge with the incoming read data.

However, due to possible skew between the incoming DQS and data it is necessary to further adjust the delay units used for sampling of the incoming DQS. Delay can be changed from the default 1/4 cycle to any value in the range of 0 through 1/2 cycle. This process is known as the read calibration. Figure 8 illustrates this concept.

Calibration sequence is done by performing a repeating DDR pattern with changing delay from zero up to the maximal 1/2 cycle delay. High and low failure points are registered as the delay “window edges.” The optimal offset is set to the middle value between the edges.

There are a total of four (4) read delay circuits as there is a read delay circuit per DQS signal. Upon completion of the read calibration, RDDLCTLx registers hold the absolute read delay for each DQS.

NOTE

When calibrating the read Delay line, starting configuration value must be a valid middle value (so data can be written and read in this case) though it might not be the optimal value. The Delay line calibration should be done after DQS gating and write level (for a 'fly-by' DDR3 board topology) calibrations.

13.1 Calibrating Read DQS with a Preset Delay Value

Read DQS can be calibrated without running the calibration sequence, by programming a predefined value of absolute delay value.

The source of the predefined value can be a read DQS calibration done in the past, or any other value estimated by user to optimize read DQS signal location.

Calibrating with a preset value is done by the following:

1. Write the read DQS preset value to RDDLCTL register.
2. Set MUR[FRC_MSR] bit.

13.2 Read DQS Delay Line Calibration Sequence

- Options for the data content used during DQS delay calibration:
  - Calibration with i.MX53 predefined data, written to a selected address in DDR device RAM.
- There are two options to handle the iterative DQS gating sequence:
  - Hardware DQS delay calibration sequence.
  - Software DQS delay calibration sequence.

Hardware Read DQS calibration is the recommended sequence for the typical customer board, while software sequence is only used for debug and special cases, when necessary.

It should be noted that the entire DQS read hardware calibration sequence is done using CK0, with DDR devices connected to CSD0. Applying CSD0 delay results to the DDR devices connected to CSD1 is conditioned by the a DDR board design that provides an identical route and placement to CSD0 and CSD1 DDR devices, sharing the same DQx bytes (typically done by mirror design of CSD0 and CSD1 DDR placement using both sides of the PCB).

During DQS read software calibration sequence, user can select any address for the write and read operations, so devices connected to CSD1 can participate similarly to CSD0 devices.

13.2.1 Calibration Sequence Setup With Predefined Data Content

In case predefined mode is used, (PDCMPR2[MPR_CMP]) is cleared, then the following steps should be executed:

1. Issue two resets to the read data FIFO through the DGCTRL0 register by writing to the RST_RD_FIFO bit (DGCTRL0[RST_RD_FIFO]=1) polling this bit to clear for completion of reset, and then repeat this sequence one more time. This sequence must be performed twice in order to properly reset the read data FIFO. Precharge all active banks (can be done through ESDSCR) as required by the standard.
2. Configure the predefined value, which reflects the value that will be written and compared through the read calibration, to PDCMPR1[PDV1, PDV2].
3. Issue write access to the external DDR device, with content matching [PDV1,PDV2]. For ESDCTL the following should be written (i= 0 to 3):
   — PDV1[7:0] to [CSD0_DDR_BA+ 0x10000000 + i * 16]
   — PDV1[15:8] to [CSD0_DDR_BA+ 0x10000004 + i * 16]
   — PDV2[7:0] to [CSD0_DDR_BA+ 0x100000008 + i * 16]
   — PDV2[15:8] to [CSD0_DDR_BA+ 0x1000000c + i * 16]

   [PDV1,PDV2] setup can be any value but empirical data shows that a predefined value of 0x00FFFF00 yields the most robust results.
4. Ensure that the initial value that is configured in the read delay line absolute offset of each byte (RDDLCTL[RD_DL_ABS_OFFSET#]) will place the read DQS somewhere inside the read DQ window. Note that the default value of 0x40 should be sufficient so no action is necessary by the user, unless the user has changed these values prior to the calibration process.
5. Start the calibration process by asserting HW_RD_DL_EN. The ESDCTL will then perform the hardware sequence as described is Section 13.2.2, "Read DQS Hardware Delay Line Calibration Sequence."
6. Poll RDDLHWCTL[HW_RD_DL_EN] until this bit clears to indicate completion. Also, check RDDLHWCTL[HW_RD_DL_ERR3, HW_RD_DL_ERR2, and HW_RD_ DL_ERR1, HW_RD_ DL_ERR0] to see if any errors occurred during calibration.
7. After hardware calibration, result DQS read delay values are valid at RDDLCTL register. Content can be read and saved for future delay setup. For debug purpose, the read DQS window high and low edges are also valid at RDDLHWST0 and RDDLHWST1.
8. Alternately, if software read delay calibration is required, go on with the user code, as described in Section 13.2.3, "DQS Read Calibration Software Sequence."

13.2.2 Read DQS Hardware Delay Line Calibration Sequence

The following steps will be executed automatically by the ESDCTL for both data modes:

1. ESDCTL waits till the read delay-line is updated with the absolute delay value for all bytes at RDDLCTL[RD_DL_ABS_OFFSETx] and also satisfying the Tmod + 4 requirement.

2. ESDCTL drives read command to the external DDR devices and waits 16 or 32 cycles (according to RDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device.
3. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS isn't inside the read DQ window and the ESDCTL generates an error for the associated byte at RDDLHWCTL[HW_RD_DL_ERRx]. If the comparison passes then ESDCTL advances to next step.
4. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
5. ESDCTL decrements the read delay line absolute offset of each byte by 1 (RDDLCTL[RD_DL_ABS_OFFSETx]) and issues a measurement process of the read delay-line to update itself with the new value.
6. ESDCTL drives read command to the DDR external devices and waits 16 or 32 cycles (according to RDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device.
7. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low read boundary of the associated byte for each byte at RDDLHWST0/1[HW_RD_DL_LOWx]. If the comparison passes then ESDCTL repeats steps 4-6. If all read data comparisons fail then the ESDCTL advances to the next step.
8. The ESDCTL starts seeking the upper boundary and sets the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issues a measurement process of the read delay-line to update itself with the new value.
9. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
10. ESDCTL drives read command to the DDR external devices and waits 16 or 32 cycles (according to RDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device.
11. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper read boundary of the associated byte for each byte at RDDLHWST0/1[HW_RD_DL_UPx]. If the comparison passes then ESDCTL increments the read delay line absolute offset of each byte by 1 (RDDLCTL[RD_DL_ABS_OFFSETx]) and issues a measurement process of the read delay-line to update itself with the new value.
12. If all read data comparisons fail then the ESDCTL advances to the next step. otherwise, ESDCTL repeats steps 9-11.
13. After the ESDCTL finds the window boundary (lower and upper) of each read data byte then it stores the average between lower and upper boundaries at the associated RDDLCTL[RD_DL_ABS_OFFSETx] and issues a measurement process of the read delay-line to update itself with the new value.

14. ESDCTL indicates that the read data calibration has finished by setting RDDLHWCTL[HW_RD_DL_EN] = 0.

13.2.3 DQS Read Calibration Software Sequence

The following steps will be executed manually by software for both modes (predefined value):

1. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.
2. Wait 16 DDR cycles till the read delay-line is updated with the absolute delay value for all bytes.
3. Issue read command (with any legal DDR address) from the external DDR device.
4. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS is not inside the read DQ window. If the comparison passes then advance to next step.
5. Reset the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
6. Decrement the read delay line absolute offset of each byte by 1 (RDDLCTL[RD_DL_ABS_OFFSETx]).
7. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.
8. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device and wait 16 or 32 cycles (according to RDDLHWCTL[HW_RD_DL_CMP_CYC]) assuming that the data has arrived from the DDR device.
9. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low read boundary of the associated byte at of each byte. If the comparison passes then repeat steps 5-8. If all read data comparisons fail then advance to the next step.
10. Start seeking the upper boundary and set the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4.
11. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.
12. Reset the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
13. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device.
14. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the upper read boundary of the associated byte at of each byte. If the comparison passes then increment the read delay line absolute offset of each byte by 1 (RDDLCTL[RD_DL_ABS_OFFSETx]).
15. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.
16. If all read data comparisons fail then advance to the next step, else repeat steps 12-15.

17. After finding the window boundary (lower and upper) of each read data byte then calculate the average between lower and upper boundaries and store the associated average at RDDLCTL[RD_DL_ABS_OFFSETx].
18. Force the delay line to measure itself and to issue the requested read delay by configuring MUR[FRC_MSR] = 1.

14 Write DQS Delay Calibration

Calibrating the write DQS delay line is used to adjust the write-DQS within the write-data byte. When write-data is output from the ESDCTL, the associated write-DQS rising and falling edges are centered-aligned with the write-data.

However, due to possible skew between the output DQS and data, it is necessary to further adjust the delay units used for delaying the sampling of the output DQS. Delay can be changed from the default 1/4 cycle to any value in the range of 0 through 1/2 cycle. This process is known as the read calibration. Figure 9 illustrates this concept.

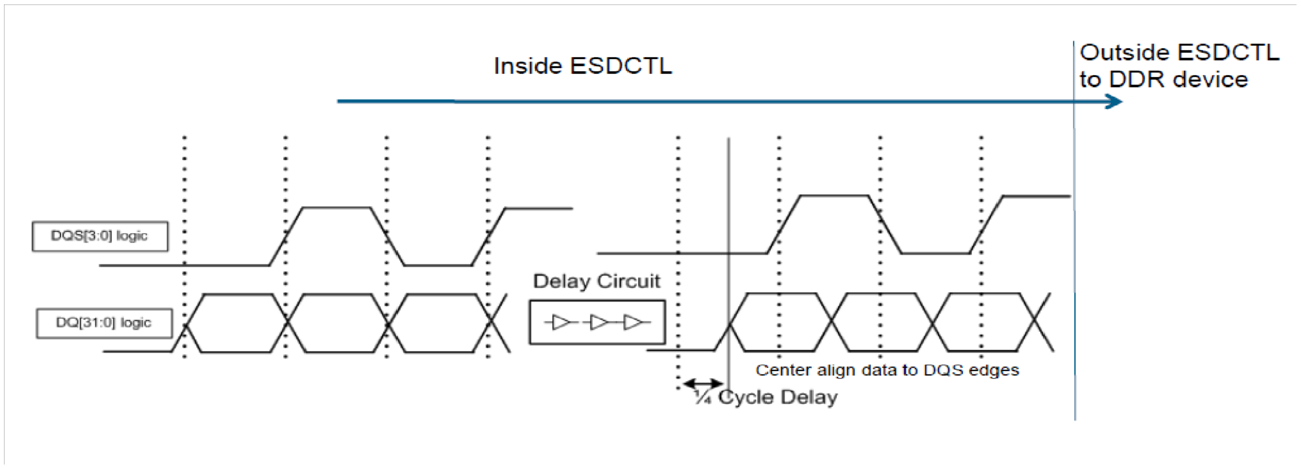

Figure 9. Illustration of Data Being Delayed to Center Align it with DQS

Calibration sequence is done by performing a repeating DDR pattern with changing delay from zero up to the maximal 1/2 cycle delay. High and low failure points are registered as the delay “window edges”. The optimal offset is set to the middle value between the edges.

There are a total of four (4) write delay circuits as there is a write delay circuit per DQS signal. Upon completion of the read calibration, WRDLCTLx registers hold the absolute read delay for each DQS.

NOTE

When calibrating the write delay line, starting configuration value must be a valid middle value (so that the data can be written and read in this case), though it might not be the optimal value. The write delay line calibration should be done after write level (for a ‘fly-by’ DDR3 board topology), DQS gating and read delay calibrations.

14.1 Calibrating Write DQS with a Preset Delay Value

Write DQS can be calibrated without running the calibration sequence, by programming a predefined value of absolute delay.

The source of the predetermined value can be a write DQS calibration done in the past, or any other value estimated by user to optimize write DQS signal location.

Calibrating with a preset value is done by the following:

1. Write the write DQS preset value to WRDLCTL register.
2. Set MUR[FRC_MSR] bit.

14.2 Write DQS Delay Line Calibration Sequence

- The options to set the data content used during DQS delay calibration:
  - Calibration with i.MX53 predefined data, written to a selected address in DDR device RAM.
- There are two options to handle the iterative DQS gating sequence:
  - Hardware Write DQS delay calibration sequence.
  - Software Write DQS delay calibration sequence.

Hardware Write DQS calibration is the recommended sequence for the typical customer board, while software sequence is only used for debug and special cases, when necessary.

It should be noted that the entire DQS write hardware calibration sequence is done using CK0, with DDR devices connected to CSD0. Applying CSD0 delay results to the DDR devices connected to CSD1 is conditioned by the DDR board design that provides an identical route and placement to CSD0 and CSD1 DDR devices, sharing the same DQx bytes (typically done by mirror design of CSD0 and CSD1 DDR placement using both sides of the PCB).

During DQS write software calibration sequence, user can select any address for the write and read operations, so devices connected to CSD1 can participate similarly to CSD0 devices.

14.2.1 Calibration Setup with Predefined Data Content.

In case predefined mode is used, (PDCMPR2[MPR_CMP]) is cleared, then the following steps should be executed:

1. Issue two resets to the read data FIFO through the DGCTRL0 register by writing to the RST_RD_FIFO bit (DGCTRL0[RST_RD_FIFO]=1) polling this bit to clear for completion of reset, and then repeat this sequence one more time. This sequence must be performed twice in order to properly reset the read data FIFO.
2. Precharge all active banks (can be done through ESDSCR) as required by the standard.
3. Configure the predefined value, which reflects the value that will be written and compared through the read calibration, to PDCMPR1[PDV1,PDV2].
4. Issue write access to the external DDR device, with content matching [PDV1,PDV2]. For ESDCTL the following should be written (i= 0 to 3):
   - PDV1[7:0] to [CSD0_DDR_BA+ 0x10000000 + i * 16]
   - PDV1[15:8] to [CSD0_DDR_BA+ 0x10000004 + i * 16]
   - PDV2[7:0] to [CSD0_DDR_BA+ 0x100000008 + i * 16]
   - PDV2[15:8] to [CSD0_DDR_BA+ 0x1000000c + i * 16]

   [PDV1,PDV2] setup can be any value but empirical data shows that a predefined value of 0x00FFFF00 yields the most robust results.

5. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (WRDLCTL[WR_DL_ABS_OFFSET#]) will place the write DQS somewhere inside the write DQ window. Note that the default value of 0x40 should be sufficient so no action is necessary by the user, unless the user has changed these values prior to the calibration process.
6. Start the calibration process by asserting HW_WR_DL_EN. The ESDCTL will then perform the hardware sequence as described is Section 14.2.2, "Write DQS Hardware Delay Line Calibration Sequence."
7. Poll WRDLHWCTL[HW_WR_DL_EN] until this bit clears to indicate completion. Also, check WRDLHWCTL[HW_WR_DL_ERR3, HW_WR_DL_ERR2, HW_WR_ DL_ERR1, HW_WR_ DL_ERR0] to see if any errors occurred during calibration.
8. Result DQS write delay values are now valid at WRDLCTL register. The contents can be read and saved for future delay setup. In addition, for debug information, the upper and lower boundaries for each byte found during the calibration process can be found in the WRDLHWST0 and WRDLHWST1 registers.
9. Alternately, if software write delay calibration is required, go on with the user code, as described in Section 14.2.3, "Write DQS Software Delay Line Calibration Sequence."

14.2.2 Write DQS Hardware Delay Line Calibration Sequence

The following steps will be executed automatically by the ESDCTL for both data modes (predefined value):

1. ESDCTL waits till the write delay-line is updated with the absolute delay value for all bytes at WRDCTL[WR_DL_ABS_OFFSETx].
2. ESDCTL drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to WRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device.
3. ESDCTL drives read command to the same address from the external DDR
4. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS isn't inside the write DQ window and the ESDCTL generates an error for the associated byte at WRDLHWCTL[HW_WR_DL_ERRx]. If the comparison passes then ESDCTL advances to next step.
5. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
6. ESDCTL decrements the write delay line absolute offset of each byte by 1 (WRDLCTL[WR_DL_ABS_OFFSETx]) and issues a measurement process of the write delay-line to update itself with the new value.
7. ESDCTL drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to WRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device.
8. ESDCTL drives read command to the same address from the external DDR.

9. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low write boundary of the associated byte of each byte at WRDLHWST0/1[HW_WR_DL_LOWx]. If the comparison passes then ESDCTL repeats steps 5-8. If all data comparisons fail then the ESDCTL advances to the next step.
10. The ESDCTL starts seeking the upper boundary and sets the write delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issues a measurement process of the write delay-line to update itself with the new value.
11. ESDCTL resets the read FIFO (to the inverted predefined value) and its pointers by setting DGCTRL[RST_RD_FIFO] = 1.
12. ESDCTL drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to WRDLHWCTL[HW_WR_DL_CMP_CYC]) assuming that the data has arrived to the DDR device.
13. ESDCTL drives read command to the same address from the external DDR.
14. ESDCTL compares the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper write boundary of the associated byte of each byte at WRDLHWST0/1[HW_WR_DL_UPx]. If the comparison passes then ESDCTL increments the write delay line absolute offset of each byte by 1 (WRDLCTL[WR_DL_ABS_OFFSETx]) and issues a measurement process of the write delay-line to update itself with the new value.
15. ESDCTL repeats steps 11-14. If all data comparisons fail then the ESDCTL advances to the next step.
16. After the ESDCTL finds the window boundary (lower and upper) of each write data byte then it stores the average between lower and upper boundaries at the associated WRDLCTL[WR_DL_ABS_OFFSETx] and issues a measurement process of the write delay-line to update itself with the new value.
17. ESDCTL indicates that the write data calibration had finished by setting WRDLHWCTL[HW_WR_DL_EN] = 0.

14.2.3 Write DQS Software Delay Line Calibration Sequence

The following steps will be executed manually by software:

1. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (WRDLCTL[WR_DL_ABS_OFFSETx]) will place the write DQS somewhere inside the write DQ window.
2. Configure the predefined value, which reflects the value that will be written and compared through the write calibration, to PDCMPR1[PDV1, PDV2].
3. Force the delay line to measure itself and to issue the requested write delay by configuring MUR[FRC_MSR] = 1.
4. Wait 16 DDR cycles till the write delay-line is updated with the absolute delay value for all bytes.
5. Issue write command to any legal DDR address of the external DDR device.
6. Issue read command, to the address written previously, from the external DDR device.

7. Compare the read data byte to the associated byte in the predefined value for all bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS is not inside the write DQ window. If the comparison passes then advance to next step.
8. ESDCTL resets the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
9. Decrement the write delay line absolute offset of each byte by 1 (WRDLCTL[WR_DL_ABS_OFFSETx]).
10. Force the delay line to measure itself and to issue the requested write delay by configuring MUR[FRC_MSR] = 1.
11. Issue write command to any legal DDR address of the external DDR device.
12. Issue read command, to the address written previously, from the external DDR device.
13. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low write boundary of the associated byte of each byte at WRDLHWST0/1[HW_WR_DL_LOWx]. If the comparison passes then repeat steps 8-12. If all data comparisons fail then advance to the next step.
14. Start seeking the upper boundary and set the write delay line absolute offset of each byte to the initial value + 1.
15. Force the delay line to measure itself and to issue the requested write delay by configuring MUR[FRC_MSR] = 1.
16. Reset the read FIFO (to the inverted predefined value) and it's pointers by setting DGCTRL[RST_RD_FIFO] = 1.
17. Issue write command to any legal DDR address of the external DDR device.
18. Issue read command, to the address written previously, from the external DDR device.
19. Compare the read data byte to the associated byte in the predefined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the upper write boundary of the associated byte of each byte at WRDLHWST0/1[HW_WR_DL_UPx]. If the comparison passes then increment the write delay line absolute offset of each byte by 1.
20. Force the delay line to measure itself and to issue the requested write delay by configuring MUR[FRC_MSR] = 1.
21. If all read data comparisons fail then advance to the next step else repeat steps 16-20.
22. After finding the window boundary (lower and upper) of each write data byte then calculate the average between lower and upper boundaries and store the associated average at WRDLCTL[WR_DL_ABS_OFFSETx].
23. Force the delay line to measure itself and to issue the requested write delay by configuring MUR[FRC_MSR] = 1.

15 Write Data Bit Delay Calibration

Write fine tuning is an additional circuit that enables the option to fine tune the timing of the DQ/DM bits (relative to DQS).

i.MX53 design is optimized of the setup value of ‘0’ for all the write data bit delay fields, and this is the recommended DDR setup for all i.MX53 boards.

If, however, a trace length and skew issue is detected on a specific board design, it is possible to mitigate this by changing the bit delay accordingly. One unit delay makes a change of 30-35 ps in signal timing. It is thus possible to set a change of up to ±100 ps, using the full range of 0 to 3 delay units in the bit delay field.

There is no hardware calibration sequence to determine the optimal bit delay values. User should test for best DDR performance, trying different values of bit delay.

16 Read Data Bit Delay Calibration

Read fine tuning is an additional circuit that enables the option to fine tune the timing of the DQ/DM bits (relative to DQS).

i.MX53 design is optimized of the setup value of ‘3’ for all the read data bit delay fields, and this is the recommended DDR setup for all i.MX53 boards.

If, however, a trace length and skew issue is detected on a specific board design, it is possible to mitigate this by changing the bit delay accordingly. One unit delay makes a change of 30-35 ps in signal timing. It is thus possible to set a change of up to ±100 ps, using the full range of 0 to 6 delay units in the bit delay field.

There is no hardware calibration sequence to determine the optimal bit delay values. User should test for best DDR performance, trying different values of bit delay.

17 Clock Delay Calibration

SDCLK signal can be added 0 to 3 delay units relative to all other DDR signals. Delay unit period is derived from the internal ESDCTL clock.

With proper DDR board design, there is no need for the clock delay, which can be kept to its default 0.

In other cases, where skews are detected in actual DDR interface, clock delay can serve as a software solution to mitigate that.

- To add delay to SDCLK0, SDCLK0_B, configure SDCTRL[SDCLK0_DEL].
- To add delay to SDCLK1, SDCLK1_B, configure SDCTRL[SDCLK1_DEL].

18 CA-Bus Bit Delay Calibration

CA-bus fine tuning is an additional circuit that enables the option to fine tune the timing of the CA-Bus bits (relative to SDCLK).

i.MX53 design is optimized to the setup value of ‘0’ for all the CA-Bus bit delay fields, and this is the recommended DDR setup for all i.MX53 boards.

If, however, a trace length and skew issue is detected on a specific board design, it is possible to mitigate this by changing the bit delay accordingly. One unit delay makes a change of 30-35 ps in signal timing. It is thus possible to set a change of up to 100 ps, using the full range of 0 to 3 delay units in the bit delay field.

To add CA-Bus delay, configure WRCADL[WR_CAx_DEL] register field with a proper value.

There is no hardware calibration sequence to determine the optimal bit delay values. User should test for best DDR performance, trying different values of bit delay.

19 DDR Calibration Code Examples

19.1 DQS Gating, Write and Read Delay Code Example

```
void ddr_calibration (void){

    int PDDWord  = 0x55aaaa55;

    int PDD55    = 0x55555555;

    int PDDAA    = 0xAAAAAAAA;

    float DQSDelayID0,DQSDelayID1,DQSDelayID2,DQSDelayID3;

    float DQS2DQDelayID0,DQS2DQDelayID1,DQS2DQDelayID2,DQS2DQDelayID3;

    int Offset,OffsetID0,OffsetID1,OffsetID2,OffsetID3;

    WORD CheckAddr,WriteData1,WriteData2,ReadData1,ReadData2;

    int i;



    reg32_write(DDR_PHY_PDCMPR1,PDDWord);

//***************************************************************

//DQS gating calibration

//***************************************************************


    for (i=0;i<4;i=i+1){

       reg32_write(CSD0_DDR_BASE_ADDR+0x10000000+i*16,PDD55);

       reg32_write(CSD0_DDR_BASE_ADDR+0x10000004+i*16,PDDAA);

       reg32_write(CSD0_DDR_BASE_ADDR+0x10000008+i*16,PDDAA);

       reg32_write(CSD0_DDR_BASE_ADDR+0x1000000C+i*16,PDD55);

    }


    reg32setbit(DDR_PHY_DGCTRL0,30);

    reg32setbit(DDR_PHY_DGCTRL0,28);
```

```
//polling
   while (reg32_read(DDR_PHY_DGCTRL0) & 0x10001000);

   // reset
   reg32_write(DDR_PHY_DGCTRL0, reg32_read(DDR_PHY_DGCTRL0) | 0x80000000);
   while (reg32_read(DDR_PHY_DGCTRL0) & 0x80000000);

   CheckAddr  = CSD0_DDR_BASE_ADDR+0x1100000;
   WriteData1 = 0xf0f0f0f0;
   WriteData2 = 0xc3c3c3c3;
   reg64_write(CheckAddr,WriteData1,WriteData2);
   reg64_read (CheckAddr,&ReadData1,&ReadData2);
   word_compare_results(0x1,WriteData1,ReadData1);
   word_compare_results(0x2,WriteData2,ReadData2);



   //**************************************************************
   //RD calibration
   //**************************************************************
   for (i=0;i<4;i=i+1){
      reg32_write(CSD0_DDR_BASE_ADDR+0x10000000+i*16,PDD55);
      reg32_write(CSD0_DDR_BASE_ADDR+0x10000004+i*16,PDDAA);
      reg32_write(CSD0_DDR_BASE_ADDR+0x10000008+i*16,PDDAA);
      reg32_write(CSD0_DDR_BASE_ADDR+0x1000000C+i*16,PDD55);
   }

   reg32_write(ESDCTL_ESDSCR,0x40008050);  //PRECHARGE ALL THE DDR
   reg32_write(DDR_PHY_RDDLHWCTL,0x00000030);

   //polling
   while (reg32_read(DDR_PHY_RDDLHWCTL) & 0x0000001f);
```

```
// reset
reg32_write(DDR_PHY_DGCTRL0, reg32_read(DDR_PHY_DGCTRL0) | 0x80000000);
while (reg32_read(DDR_PHY_DGCTRL0) & 0x80000000);

CheckAddr  = CSD0_DDR_BASE_ADDR+0x1000000;
WriteData1 = 0x01234567;
WriteData2 = 0x89ABCDEF;
reg64_write(CheckAddr,WriteData1,WriteData2);
reg64_read (CheckAddr,&ReadData1,&ReadData2);
word_compare_results(0x1,WriteData1,ReadData1);
word_compare_results(0x2,WriteData2,ReadData2);

//************************************************************
//WR calibration
//************************************************************

for (i=0;i<4;i=i+1){
   reg32_write(CSD0_DDR_BASE_ADDR+0x10000000+i*16,PDD55);
   reg32_write(CSD0_DDR_BASE_ADDR+0x10000004+i*16,PDDAA);
   reg32_write(CSD0_DDR_BASE_ADDR+0x10000008+i*16,PDDAA);
   reg32_write(CSD0_DDR_BASE_ADDR+0x1000000C+i*16,PDD55);
}

 reg32_write(ESDCTL_ESDSCR,0x40008050);  //PRECHARGE ALL THE DDR
reg32_write(DDR_PHY_WRDLHWCTL,0x00000030);

//polling
while (reg32_read(DDR_PHY_WRDLHWCTL) & 0x0000001f);

// reset
reg32_write(DDR_PHY_DGCTRL0, reg32_read(DDR_PHY_DGCTRL0) | 0x80000000);
while (reg32_read(DDR_PHY_DGCTRL0) & 0x80000000);


CheckAddr  = CSD0_DDR_BASE_ADDR+0x1000000;
```

```
    WriteData1 = 0x01234567;
    WriteData2 = 0x89ABCDEF;
    reg64_write(CheckAddr,WriteData1,WriteData2);
    reg64_read (CheckAddr,&ReadData1,&ReadData2);
    word_compare_results(0x1,WriteData1,ReadData1);
    word_compare_results(0x2,WriteData2,ReadData2);



    //********************************************************
    // Print DQS gating, DQS read/write delay values:
    //********************************************************
    printf("0x63fd907c: %x\n", reg32_read(0x63fd907c));
    printf("0x63fd9080: %x\n", reg32_read(0x63fd9080));
    printf("0x63fd9088: %x\n", reg32_read(0x63fd9088));
    printf("0x63fd9090: %x\n", reg32_read(0x63fd9090));
```

19.2 Write Leveling Code Example

```
void write_leveling()
{

   //increase RALAT, WALAT to max:
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ESDMISC, 0x000b17c0);

   // Disable ZQ calibration auto init, to avoid interfering with WL calibration:
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ZQHWCTRL, 0x0);

   // Disable Auto refresh, to avoid interfering with WL calibration:
   reg32_write((ESDCTL_BASE_ADDR + ESDCTL_ESDREF),0x0000C000);

   //precharge all CS0 DDR:
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ESDSCR, 0x40008050);

   //set mem device of CS0 to writing leveling
   //+ enable DQS output by setting WL_EN bit
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ESDSCR, 0x00c28231);
```

```
    // start HW write leveling process at DDR_PHY:
    reg32_write(DDR_PHY_WLGCR, reg32_read(DDR_PHY_WLGCR) | 0x00000001);

    //wait for write leveling HW termination with no error:
    while(reg32_read(DDR_PHY_WLGCR) & 0x00000f01)

   //clear mem device of CS0 writing leveling
   //+ disable DQS output by setting WL_EN bit
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ESDSCR, 0x00428031);

   // Enable ZQ calibration auto init:
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ZQHWCTRL, 0x04b80003);

  // Enable Auto refresh, to aviod interfering with WL calibration:
   reg32_write(ESDCTL_BASE_ADDR + ESDCTL_ESDREF,0x00001800);

  //Print WL delay value registers:
   printf("0x63fd904c: %x\n", reg32_read(0x63fd904c));
   printf("0x63fd9050: %x\n", reg32_read(0x63fd9050));

  //Print WL error register:
   printf("0x63fd90b8: %x\n", reg32_read(0x63fd90b8));
}
```

20 References

The following are Freescale documents available at www.freescale.com:

- *i.MX53 Multimedia Applications Processor Reference Manual* (IMX53RM)
- *i.MX53 System Development User's Guide* (MX53UG)

Additional references:

- JEDEC Solid State Technology Association, *DDR3 SDRAM Standard* (JESD79-3D). Arlington, VA, USA: 2009.
- JEDEC Solid State Technology Association, *Low Power Double Data Rate 2 (LPDDR2)* standard. Arlington, VA, USA: 2011.

21 Revision History

Table 5 provides a revision history for this application note.

Table 5. Document Revision History

Rev. Number	Date	Substantive Change(s)
1	04/2013	• Section 10.4, "Local ESDCTL PHY Hardware ZQ Calibration Sequence": Updated ZQ description, added ZQ pad name, updated ZQ resistor value. • Section 11.2, "Hardware Write Leveling Calibration Sequence," step 11, changed "factional" to "functional." • Section 12.3.1, "Register Setup Prior to DQS Gating Calibration Sequence: Changed title of Table 3 to "Auto Initiated Hardware ZQ Calibration Configurations." • Section 19, "DDR Calibration Code Examples": Changed precharge value from 0x04000050 to 0x40008050. • Throughout: minor editing for clarification.
0	02/02012	Initial release.

Document Number: AN4466
Rev. 1
05/2013