Freescale Semiconductor
Application Note

Document Number: AN3992
Rev. 0, 02/2010

i.MX51 DDR/mDDR Calibration Procedure

by *Multimedia Application Division*
Freescale Semiconductor, Inc.
Austin, TX

This application note describes the calibration procedure to find the optimal delay line settings for the i.MX51, to work with Mobile DDR (double data rate) or DDR2 (extended version of DDR) memories. These delay line settings determine the time at which each of the DQS (data query strobe) lines sample its corresponding data lines.

Optimal delay line settings are determined by comparing the corresponding delay values of each board reference design or product, (as different delays and capacitances are introduced and different memory devices are used) to have more or less the same values. This value is used across the same type of board package or product.

NOTE

It is recommended to read the DDR controller (ESDCTLv2) chapter in the *MCIMX51 Multimedia Applications Processor Reference Manual* (*MCIMX51RM)*, to get familiarized with the ESDCTLv2's functionality and programming model.

Contents

freescale™ semiconductor

1 Introduction

The goal of the calibration procedure is to align the DQS edges, to sample the data at the mid point of the steady data window (which changes with respect to the DDR clock period rate). This DQS alignment is required for both read and write operations.

DQS gating calibration is required for DDR2 memories as they use differential DQS lines with on-die termination (ODT) and do not have on board pull-up or pull-down resistors on the DQS lines. During a read operation, if a DQS line is not driven, a high impedance value can propagate through the DDR controller and can be misinterpreted as an access. Therefore, the DQS signals should be gated internally and used only when needed. This is done by the DQS calibration procedure.

For mDDR (mobile DDR) memories, pull-down resistors can be used on the DQS lines. Therefore, DQS lines always have a defined value and need not be gated.

NOTE

It is recommended to perform the calibration with VCC at 1.2 V.

2 DDR2 Calibration Procedure

The steps required for calibration while using DDR2 memories are:

1. Set DDR2 to single-ended mode.
2. Perform calibration for write and read delay parameters.
3. Set DDR2 to differential mode.
4. Perform calibration for the DQS gating.
5. Perform the calibration again for write and read delay parameters (as in step 2).

For MDDR memories, perform only the calibration for write and read delay parameters.

3 Delay Line Calibration

The concept of the delay calibration is similar for both read and write. The main difference is that, for read cycle, the DQS to DQ timing is relative to the i.MX51 internally. For the write cycle, the DQS to DQ timing is relative to the DDR memory device. As shown in Figure 1, the goal is to find the center of the valid sample window which meets the setup and hold time requirements.

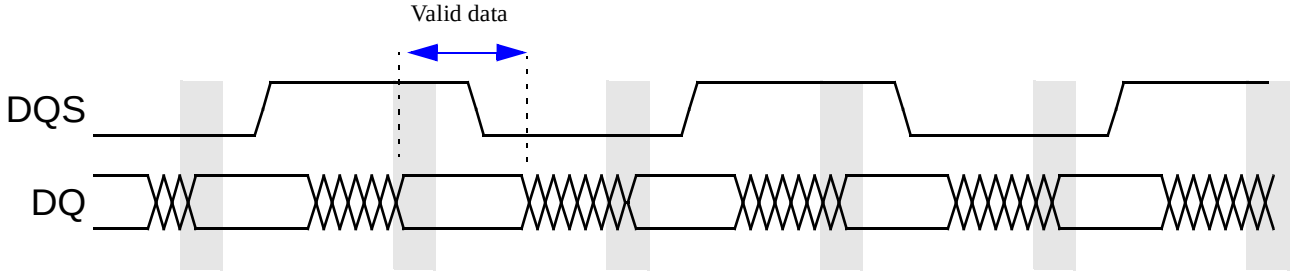

Figure 1. DQ and DQS Timing

Figure 2 shows the magnified view of DQ and DQS timing.

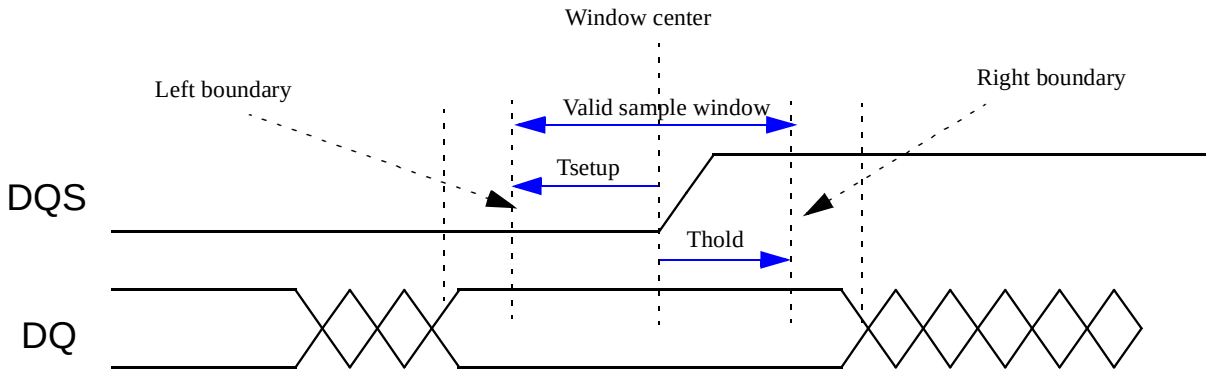

Figure 2. Magnified View of DQ and DQS Timing

3.1 Delay Line Registers

There are five DQS lines (four for read and one for write) and hence there are five delay line registers in the ESDCTLv2 (DDR controller) module:

- Four registers control the delay of the DQS signal for read.
- One register controls the delay of the data bus for write.

The register names, its address locations, and its corresponding functions are as follows:

- Register ESDCDLY1—at address 0x83fd9020—controls the delay of DQS[0] for read data bits[7:0].
- Register ESDCDLY2—at address 0x83fd9024—controls the delay of DQS[1] for read data bits[15:8].
- Register ESDCDLY3—at address 0x83fd9028—controls the delay of DQS[2] for read data bits[23:16].
- Register ESDCDLY4—at address 0x83fd902c—controls the delay of DQS[3] for read data bits[31:24].
- Register ESDCDLY5—at address 0x83fd9030—controls the delay of all the data bus for write data bits[31:0].

The delay line register is configured using the following two parameters:

- DLY_ABS_OFFSET_#—Amount of delay added, normalized by one DDR clock period. DLY_ABS_OFFSET_# maintains the same delay between DQS and data by taking into account of the changes in process and operating conditions of the i.MX51 device. For example, if a value of 128 units (¼ cycle) is programmed, then the DLY_ABS_OFFSET_# includes the amount of delay units that is needed to provide a ¼ cycle delay. The number of delay units is automatically changed depending on the temperature, voltage and type of silicon case used, to keep a constant delay of ¼ of DDR clock cycle.
- DLY_OFFSET_#—Amount of physical delay units added. The delay of the units vary between different parts and across different operating conditions.

3.2 Calculation of Delay Parameters for the Delay Lines

The delay for each delay line is calculated according to Equation 1:

$$Total\_delay = (measure \times DLY\_ABS\_OFFSET\_5) \div 512 + DLY\_OFFSET\_5 \quad \textbf{Eqn. 1}$$

where:

- measure—Amount of delay units in one DDR clock period. This parameter is calculated by the hardware.
- 512—Represents the maximum number of time steps in one DDR clock period.
- DLY_ABS_OFFSET_#—Set to 128 by default which is equivalent to ¼ cycle.
- DLY_OFFSET_#—Set to –12 by default to compensate for the constant delay of 12 units.
- Total_delay—Represents the number of delay units in the write delay line. It takes into account the measurement that is performed by the measurement unit and adds the delay programmed in the ESDCDLY5 register. The value of *Total_delay* is represented by the ESDGPR[7:0] register (bit field is called QTR_CYCLE_LENGTH) at address 0x83fd9034.

The *measure* value can be found by a simple calculation from Equation 1 and it is used for both read and write delay calculations. The delay for the four read delay lines follows the same formula—Equation 1, using the corresponding parameters from the ESDCDLY1, ESDCDLY2, ESDCDLY3, and ESDCDLY4 registers. The total delay (*Total_delay* parameter) is reflected only for the write delay line.

The recommended method to find the calibration parameters for the write and read delay lines is to find the optimal DLY_ABS_OFFSET values, without using the DLY_OFFSET (remains at default of –12 unit). This reduces the sensitivity of changes to process voltage temperature (PVT) conditions, as the delays are defined as fractions of a cycle.

NOTE

The value of DLY_ABS_OFFSET_# is not directly translated into the number of physical delay units. It depends on the silicon speed and the DDR frequency. The maximum number of delay units per cycle while running at 100 MHz on best case silicon is 512. This means that at 200 MHz and on typical silicon, approximately 2–5 consecutive values of DLY_ABS_OFFSET_# may provide the same number of delay units.

By default, a constant delay of 12 units exist between each DQS lines and the data lines, for both read and write. This is an internal delay in the design and cannot be removed.

3.2.1 DLY_ABS_OFFSET_#

The DLY_ABS_OFFSET_bits[15:8] allow values in the range of [0:255]. But the total equations (Total_delay) are bound between 0 (that means 12 delay units which is the minimum) and half the DDR clock period delay.

The example below explains how the DLY_ABS_OFFSET_# value is transformed to delay in silicon.

If

- *Total_delay* result that is reflected by ESDGPR[7:0] bits is 24.
- The DDR frequency is set to 200 MHz (5 ns period).
- The *DLY_OFFSET_5* is set to its default value (that is –12).
- The *DLY_ABS_OFFSET_5* is set to its default value (that is 128).

$$measure = 512 \times (Total\_delay + DLY\_OFFSET\_5) \div DLY\_ABS\_OFFSET\_5 \qquad \textbf{Eqn. 2}$$

Then, *measure* result is 144, which means that each delay unit adds about 35 ps in silicon (period ÷ measure = 5 ÷144 = 35 ps (app)).

The amount of delay introduced due to *DLY_ABS_OFFSET_5* is (144) × (128 ÷ 512) = 36 delay units. In case, this delay is 12 units or lesser, *Total_delay* value is read as 0 which corresponds to the minimum delay applied by the constant 12 units.

From this point onwards, wherever delay, delay window, delay line setting is mentioned, it refers to the DLY_ABS_OFFSET value (bits [15:8] of the ESDCDLY1, ESDCDLY2, ESDCDLY3, ESDCDLY4, and ESDCDLY5 registers).

3.3 Delay Line Calibration Procedure

The calibration procedure contains the steps for finding the write and read values using a check function.

NOTE

The code for the calibration procedure must be placed in the internal memory and should not be placed in the external memory (DDR), since delay line values have to be configured.

3.3.1 Check Function

The check function has the following roles:

- Gives an indication whether the current delay line settings are valid or not (validity of programmed delay values are done by a generic function that accesses the DDR, both CS0 and CS1). The test executed by this function should be stressful enough to give a good indication whether the delay line settings are valid. But, it should not take too long since this function is called several thousands of times in the calibration procedure. For more information, see Section 3.3.2.1, “Finding Write Delay Value,” and Section 3.3.2.2, “Finding Read Delay Value.”
- Perform write bursts, single writes, read bursts, single reads, 8/16/32 bit accesses and access different address spaces. Other masters, such as SDMA can also be used to stress the DDR.

- Write the data both to the DDR and to another address space in the chip (for example, i-RAM, GMEM). In this way, when the data is read from the DDR, it can be compared to the golden data stored in the internal memory.

The check function has one input (mask input) and one output (pass or fail output). The mask input is used to determine which byte (byte0, byte1, byte2 or byte3) is to be compared. Since each read delay line register controls one byte of data, only one byte can be checked at a time. The status returns PASS, if all the data that is read back from the DDR passes the comparison with the golden data. If there is at least one data (data that is checked with the relevant mask) which is incorrect, then the function returns FAIL.

The check function has the following format:

```
check_ddr(int mask)

.....

return status
```

There are two types of check functions:

- Aggressive check function—This function is used to find the actual delay values. It has access to a wide address range.
- Less aggressive function—This function is used for a quick search to find the initial values that are used by the aggressive function. This function is also called as typical check function, which has reduced access to a narrower address range.

3.3.2 Procedure for Calibration

First, set the bit 11 to 1‘b1 in the ESDMISC register (0x83fd9010), before performing the procedural steps. The FRC_MSR bit requests the delay line circuitry to constantly perform the measurements and to calculate the delay line values based on the parameters in the ESDCDLY# registers. Therefore, during the entire calibration procedure, this bit has to be set. The procedure starts with finding the delay value for write delay line and then finding the delay line values for the read.

For DDR2 memory, the delay calibration needs to be performed when DQS lines and memory are configured to single-ended mode and not differential mode. This eliminates the dependency of memory on the DQS gating when the delay of DQS lines are fluctuated during calibration.

3.3.2.1 Finding Write Delay Value

The basic algorithm for finding the delay value for the write delay line is as follows:

1. Find point zero.
2. Search for the largest target read window, for byte0 while scanning all write delay values and byte0 read delay values.
3. Repeat step 2 for byte1, byte2 and byte3.
4. Check whether the target read windows of byte1, byte2, and byte3 are valid.
5. Find the value at the center for each target read window.

6. Find the target write window, using the read windows.
7. Check if target write window is valid.
8. Find the true write window—window where the write delay value is intended to be found.
9. Check whether the true write window is valid.
10. Find the center value at the true write window.

Each step of the algorithm mentioned above are described as follows:

1. Find the largest value of DLY_ABS_OFFSET_5 that still provides zero delay (12 constant delay units)—This is done by decreasing the value of DLY_ABS_OFFSET_5 and reading the value of Total_delay each time, until it reaches zero. This value of Total_delay is recorded as point zero. It is used as the smallest value of DLY_ABS_OFFSET_#, while searching for the left boundary of the write and read valid sample windows.
2. Search for a read window per DQS line—Starting with byte0 (for DQS0), for each read delay value of DLY_ABS_OFFSET_1, search for at least one working write delay value. If no working write delay value is found, then that value of DLY_ABS_OFFSET_1 cannot be part of the window. The procedure stated above is done as follows. For each value of DLY_ABS_OFFSET_1, starting from point zero, increase the value by 5 units each time (this saves time by not going through all values, but still provides good resolution) until it reaches the maximum value (255). While increasing, search for at least one working write value. For each set of values, invoke the typical check function with selection of byte0. The result is the read window that is used to find the write delay values in step 6. See Figure 3 for information on how a window is defined.
3. Repeat step 2 for byte1, byte2, and for byte3 (in case of 32 bit memory)—This is done using the corresponding DLY_ABS_OFFSET_# field. The result is a read delay window per byte (DQS line). These windows are used as target read windows, while searching for the read delay window in Section 3.3.2.2, “Finding Read Delay Value.”
4. Check whether the target read windows of byte1, byte2 and byte3 are valid—Check whether the read windows are wide enough to allow some margin across process corners and different boards of the same type. If the window that is found is too narrow, there may be a problem with the board design.
5. Find the value at center for each of the target read window—These are the read delay values which are used to find the write delay window.
6. Find the target write window—Program the read delay values that were found. Starting from point zero, increase the value of DLY_ABS_OFFSET_5 by 5 units (until it reaches the maximum value –255) and invoke the typical check function. Select the largest window, as shown in Figure 4. The resulting window is the target write window, which is used to find the true write window in step 8.
7. Check if the target write window is a valid sample window, as done in step 4—If it is a valid window, continue to the next step. If a valid window is not found, try the calibration on few different parts. If the issue persists on multiple parts, there may be a problem with the board design.

8. Find the true write window—Program the same read delay values as in step 6. Go through all the values in the target write window starting at start point –4, until it reaches the end point +4 (since every 5 points were skipped). Increase the value of DLY_ABS_OFFSET_5 each time by 1 and invoke the aggressive check function. This gives the true write window.
9. Check whether the true write window is a valid sample window, as done in step 4—If it is a valid window, continue to the next step. If a valid window is not found, try the calibration on few different parts. If the issue persists on multiple parts, there may be a problem with the board design.
10. The center value at the true write window—It is calculated and used as the newly calculated DLY_ABS_OFFSET_5 value for the write delay line.

Finding Target Windows for Read and Write

A target window is found by using the typical check function on the working window, which consists of large number of successful consecutive delay values as shown in Figure 3. This window is used as the target to be checked by the aggressive check function (which finds the true window).

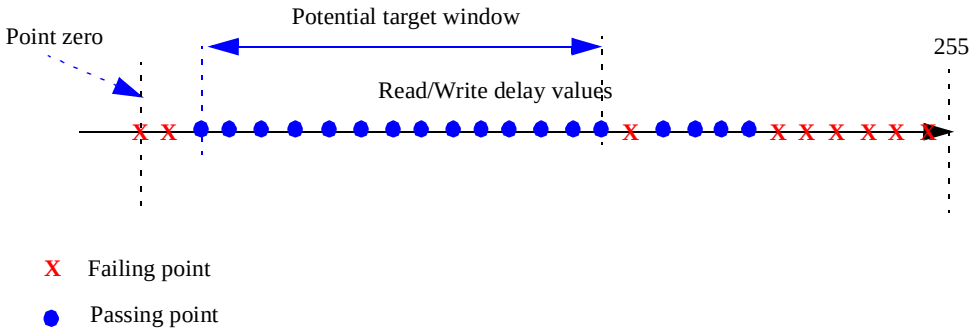

Figure 3. A Working Window

The target window is selected by taking the largest working window that is found. This is done for read (per each byte) and write, as shown in Figure 4.

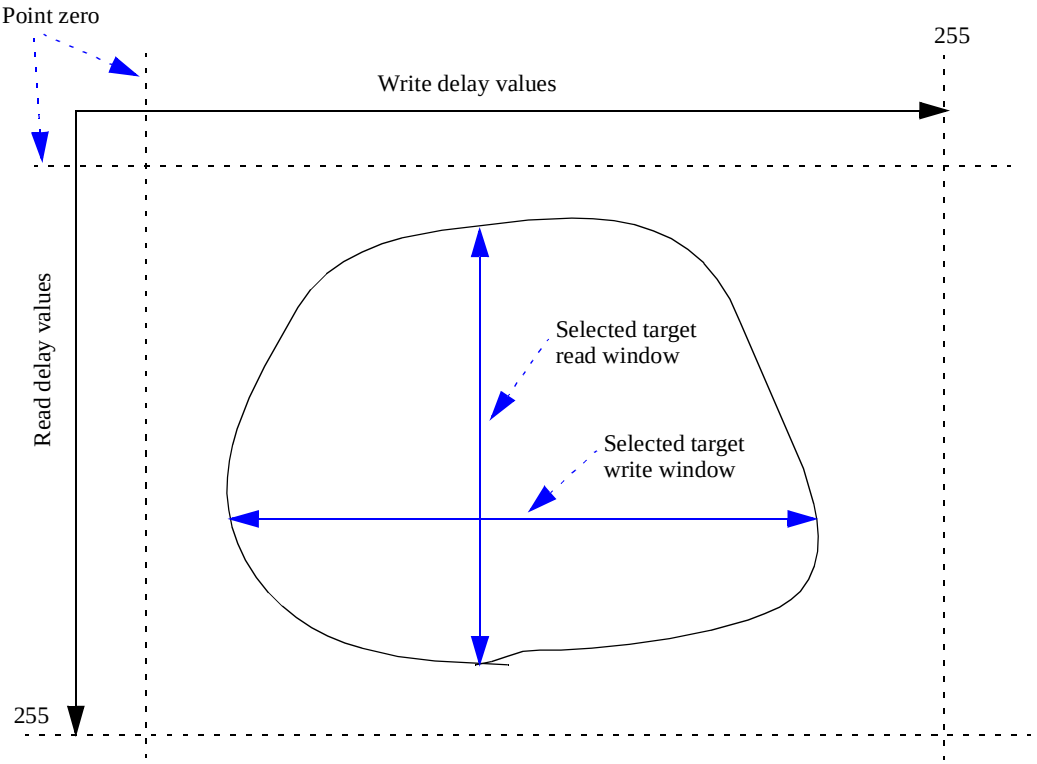

Figure 4. Selection of the Target Window

Estimated Number of Iterations

Assume that point zero is between 50–60 of DLY_ABS_OFFSET_# value. Hence, there are total of approximately 200 (out of 255) valid values. When a typical check function is used, skip 5 values each time. Therefore, approximately there are 40 values to check per each delay line.

While checking a read delay value, search write delay values, until a successful access is found. This means that maximum of 40 write delay values are searched (in case of failure).

Therefore, the maximum number of iterations, for the target read window is given by Equation 3:

$$\textit{maximum number of iterations} = 40\ (\textit{write delay}) \times 40\ (\textit{read delay}) \times \textit{number\_of\_bytes} \qquad \textbf{\textit{Eqn. 3}}$$

For 4 bytes, maximum number of iterations is as follows:

$1600 \times 4 = 6400$

For the target write window, there are 40 additional iterations. Hence, the typical function is invoked less than 6440 times.

Assuming there are around 100 values in the target write window, then the aggressive check function is invoked 100 times.

3.3.2.2 Finding Read Delay Value

The read delay values are calibrated after the write values are found and programmed in DLY_ABS_OFFSET_5. Hence, it is assumed that the data is written correctly and the failures are due to improper read delay values.

The following steps are used to find the delay values for the read delay lines:

1. Starting with byte0, go through all values from the target read window, found in step 3 of the detailed description in Section 3.3.2.1, “Finding Write Delay Value,” to find the write delay value. From the start point –4, increase the value of DLY_ABS_OFFSET_1 by 1 unit, until the end point +4 is reached. Invoke the aggressive check function with selection of byte0. This provides the true read window for DQS0.
2. Repeat step 1 for each byte.
3. Check whether the windows that are found are wide enough. If all values are valid, continue to the next step. If the windows that are found are too narrow, try the calibration on few different parts. If the issue persists on multiple parts, there may be a problem with the board design.
4. The value at the center of each window is calculated and are used as the newly calculated DLY_ABS_OFFSET_# values for the read delay lines.

Estimated Number of Iterations

Assume that there are around 100 values per each target read window, then the aggressive check function is invoked around four hundred times. After all the five delay values are known, the calibration process is completed and the FRC_MSR bit can be set back to 0.

4 DQS Gating Calibration

The purpose of the DQS gating calibration is to align the rise of the internal DQS gating signal in the *preamble period* of the incoming DQS burst. When DDR2 memories are used while using differential mode for the DQS lines, no pull-up or pull-down resistors are applied to the DQS lines. Therefore, during non-active periods, the DQS line is in a *high z* (high impedance) state. In Figure 5, the DQS is propagated as *high z* into the design since the enable signal is raised too early.

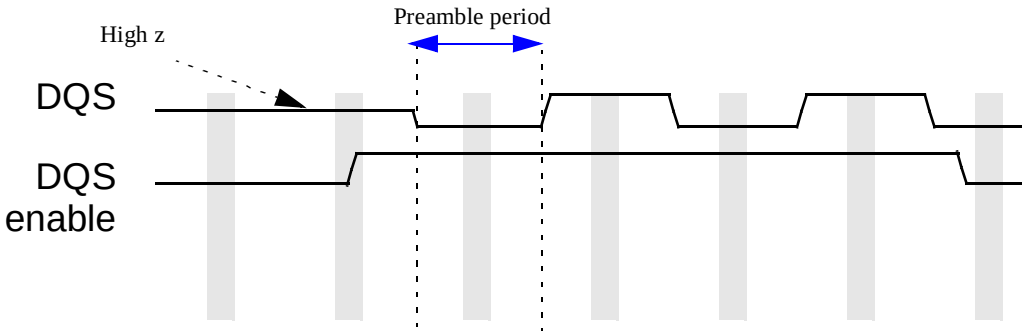

Figure 5. DQS Gating when DQS Enable Not Aligned

In Figure 6, the DQS signal is wrapped by the DQS enable signal.

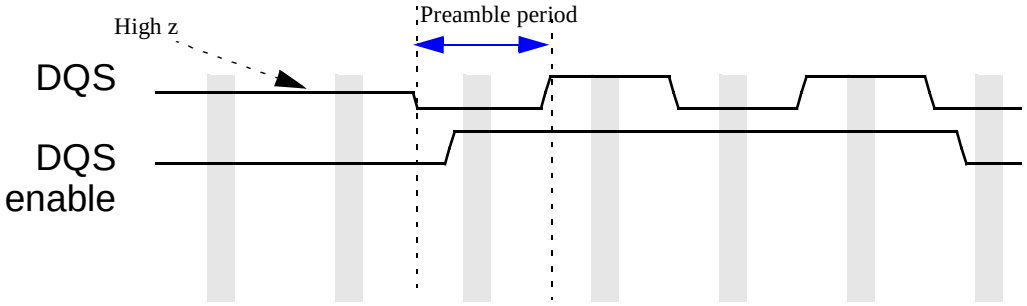

Figure 6. DQS Gating after Calibration

During the DQS gating calibration procedure, the DQS enable signal is moved around (with resolution of a quarter cycle of the DDR clock period) to get the delay that is required to properly wrap the DQS signal. The calibration is done for each of the four DQS read signals. Part of the delay is commonly configured for all. But fine tuning is done for each line separately.

4.1 DQS Gating Register

The delay parameters of the gating signals for the four read DQS lines are configured in ESDGPR (a general purpose register) register at address 0x83fd9034 of the ESDCTLv2 module.

The ESDGPR is configured as follows:

- ESDGPR[31]—DIG_EN bit—DQS in enable bit. This bit must be set while using the differential DQS signals.
- ESDGPR[30:29]—DIG_CYC—Delay in cycles, common to all four DQS lines.
- ESDGPR[28:27]—DIG_QTR—Delay in ¼ cycles, common to all four DQS lines.
- ESDGPR[26:25]—DIG_OFF0—Delay in ¼ cycles for DQS0.
- ESDGPR[24:23]—DIG_OFF1—Delay in ¼ cycles for DQS1.
- ESDGPR[22:21]—DIG_OFF2—Delay in ¼ cycles for DQS2.
- ESDGPR[20:19]—DIG_OFF3—Delay in ¼ cycles for DQS3.

Register ESDCTMISC[1] —RST bit (at address 0x83fd9034) is used to reset the ESDCTLv2 module during the calibration of the DQS gating signals.

4.2 Calculation of Delay Parameters for DQS Gating

During the calibration procedure of the DQS gating, it is assumed that the read and write delay lines are calibrated as described previously.

NOTE

The code for the calibration procedure must be placed in the internal memory and should not be placed in the external memory (DDR), since delay for the DQS enable lines are not configured.

The calibration is done by writing the expected data to the memory and performing read (read with a burst) operation each time with a different delay settings per byte (per DQS enable line). Starting from zero delay per byte, the delay is increased until the read is performed correctly. After the delay per each byte is found, an additional ¼ cycle delay is added to each DQS enable line, in order to move it further into the preamble period.

The maximum skew that can be configured between the DQS lines is 3/4 cycle. It is assumed that the required delay for each DQS enable line is nearly the same as the others and hence this skew is sufficient.

Internal variables delay0, delay1, delay2, delay3 hold ¼ cycle delay value per DQS enable line and are used for the calculations. They are incremented each time and are used to calculate the common delay and the specific offset per DQS line.

The sequence for calculating the delay parameters for DQS gating is as follows:

1. Write pre-defined data to DDR memory.
2. Set delay0, delay1, delay2, delay3 to 0.
3. Read data per each byte and according to the read value, the function to be performed is as follows:
   - If byte0 is wrong, increment delay0 by 1.
   - If byte1 is wrong, increment delay1 by 1.
   - If byte2 is wrong, increment delay2 by 1.
   - If byte3 is wrong, increment delay3 by 1.
   - If all are correct (all bytes read correct data) jump to step 7.
4. Perform a software reset—It is done by setting RST bit to 1 and setting it back to 0. This is required to reset the pointers, since the wrong reads have made the FIFO pointers to become invalid.
5. Write new values to ESDGPR register to the following fields:
   - min_delay = min(delay0, delay1, delay2, delay3)—It is the internal variable that holds the minimum delay, which is the common delay.
   - DIG_CYC = min_delay ÷ 4—It is the common delay in number of full cycles.
   - DIG_QTR = min_delay % 4—Modulo of the minimum delay gives the remaining common delay in ¼ cycles.
   - DIG_OFF0 = delay0 – min_delay—This is the remaining offset in ¼ cycles for the enable line of DQS0.
   - DIG_OFF1 = delay1 – min_delay—This is the remaining offset in ¼ cycles for the enable line of DQS1.
   - DIG_OFF2 = delay2 – min_delay—This is the remaining offset in ¼ cycles for the enable line of DQS2.
   - DIG_OFF3 = delay3 – min_delay—This is the remaining offset in ¼ cycles for the enable line of DQS3.
6. Jump to step 3.
7. Increment delay0, delay1, delay2, delay3 by 1 to move the DQS enable signals further into the preamble period.

8. Write new values to the ESDGPR register.
9. Read data (all bytes) to check if the read operation is correct.

5 Single-Ended/Differential Mode Selection

The calibration for read and write delay values should be done at single-ended mode. For the DQS gating calibration, switch to differential mode.

5.1 Setting Differential Mode

The following steps are required to operate in differential mode:

1. Set the memory to differential DQS using the LMR command.
2. Set the DQS pads of the i.MX51 to differential mode—ESDMISC register in ESDCTLv2, DIFF_DQS_EN bit.
3. Remove the pull-ups and pull-downs for the DQS and DQS_B pads by configuring the following registers in the IOMUXC:
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3.

5.2 Setting Single-Ended Mode

The following steps are required to operate in single-ended mode:

1. Set the memory to single ended DQS using the LMR command.
2. Set the DQS pads of the i.MX51 to single ended mode—ESDMISC register in ESDCTLv2, DIFF_DQS_EN bit.
3. Add pull-downs for the DQS pads by configuring the following registers in the IOMUXC:
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2.
   - IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3.
4. Disable ODT in the i.MX51 IO pads. This is done by configuring TERM_CTL0 through TERM_CTL3 fields in ESDMISC registers in the ESDCTL controller (part of EMI module).

6 Conclusion

Using the calibration procedure described in this application note, the optimal delay line settings for i.MX51 can be found. This enables the i.MX51 to work with MDDR and DDR2 memories.

7 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	02/2010	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
P.O. Box 5405
Denver, Colorado 80217
1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Document Number: AN3992
Rev. 0
02/2010